**UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO**

CENTRO UNIVERSITARIO UAEM VALLE DE MÉXICO

# Aproximación de funciones con redes neuronales y algoritmos evolutivos

**TESIS**

Que para obtener el Grado de

**MAESTRO EN CIENCIAS DE LA COMPUTACIÓN**

P r e s e n t a

**Ingeniero Alejandro Romero Herrera**

**Tutor Académico:**
**Doctor Oscar Herrera Alcántara**
**Tutor(es) Adjunto(s):**
**Doctor Víctor Manuel Landassuri Moreno**
**Doctor Asdrúbal López Chau**

**Atizapán de Zaragoza, Edo. de Méx. Marzo de 2016**

Centro Universitario
UAEM Valle de México

## Dedicatoria

A mi Familia que estuvo en todo momento en este largo camino y me apoyó en cada paso que di.

## Agradecimientos

Al Dr. Oscar Herrera Alcántara y a la Dra. Ivonne Rodríguez Pérez por creer en mí y apoyarme en todo momento para poder lograr mis objetivos.

A la Universidad Autónoma del Estado de México, al Consejo Nacional de Ciencia y Tecnología, y al Consejo Mexicano de Ciencia y Tecnología por el apoyo y las facilidades brindadas para poder llevar a cabo este trabajo de tesis y terminar mis estudios de maestría.

## Resumen

En este trabajo de investigación se aborda el problema de aproximación de funciones de energía finita, a través de un modelo de red neuronal artificial que involucra funciones wavelets y algoritmos evolutivos.

La aproximación de funciones de energía finita es un problema de interés en varias áreas del conocimiento. Por ejemplo, en la aproximación de funciones que representan variables financieras como los índices de la bolsa de valores, o variables de fenómenos naturales, como la temperatura atmosférica y la energía solar.

Los fenómenos descritos anteriormente (variables financieras y variables climáticas) por mencionar algunos, tienen en común el desconocimiento de una función explícita que las relacione con otras variables, por lo que su aproximación resulta relevante a fin de hacer estudios sobre el modelo generado.

Uno de los modelos usados para aproximar funciones consiste en la descomposición en otras funciones conocidas tales que su combinación lineal minimice el error cuadrático medio. Así, en este trabajo se propone hacer una combinación lineal de funciones wavelets vinculadas según un modelo conexionista, en donde las funciones wavelets son generadas mediante el algoritmo en cascada a partir de filtros ortogonales de reconstrucción perfecta, y a través de escalamientos y traslaciones se aproxime una función objetivo. Al modelo propuesto se le ha llamado EPWavenets, como una abreviatura de la combinación de algoritmos evolutivos, parametrización de filtros, funciones wavelets y redes neuronales artificiales.

De esta forma, se busca demostrar que es posible aproximar una función de energía finita con funciones wavelets generadas a partir de filtros paramétricos cuya combinación está determinada por la arquitectura de una red neuronal, en donde los parámetros de escalamiento y traslación, los pesos sinápticos y los parámetros de los filtros se determinan con un algoritmo evolutivo desde el punto de vista del aprendizaje supervisado.

Para demostrar lo anterior se utilizó la siguiente metodología:

1. Revisar modelos de red neuronal existentes que sean susceptibles de modificar sus funciones base y ajustar sus parámetros para aproximar funciones
2. Proponer un modelo de red neuronal que permita incluir nuevas funciones base y ajustar sus parámetros en forma supervisada
3. Identificar propiedades y familias de funciones wavelet que puedan incorporarse al modelo propuesto
4. Identificar un algoritmo evolutivo para ajustar los parámetros del modelo de red neuronal propuesto
5. Identificar un conjunto de funciones de prueba para comparar los resultados con otras redes neuronales con diferentes funciones base

6. Generar tablas comparativas y gráficas en donde se aprecie la comprobación de la hipótesis planteada a partir de los resultados experimentales

Los resultados experimentales sustentan la hipótesis, indicando que es posible usar funciones wavelets con filtros paramétricos para aproximar funciones de energía finita, en una arquitectura de red neuronal.

También se pudo confirmar que sí fue posible usar algoritmos evolutivos en *EPWavenets* para optimizar los parámetros libres, a efecto de minimizar el error de aproximación.

Derivado de los resultados se concluye que las *EPWavenets* logran un alto grado de adaptabilidad y un desempeño competitivo respecto a otras redes neuronales que involucran funciones de base radial sobre un conjunto de funciones de prueba.

## Abstract

The problem of finite energy function approximation is tackled in this research through an artificial neural network model involving wavelets functions and evolutionary algorithms.

The approximation of finite energy functions is a problem of interest in several areas of knowledge. For example, in the approximation of functions by representing financial variables as the indexes of the stock market, natural phenomena or variables such as atmospheric temperature, and solar energy.

The phenomena described above (financial variables and climatic variables) to mention a few of them, have in common the lack of an explicit function that relates to other variables, so their approximation is relevant for our research purpose on the generated model.

One of the models used to approximate functions is to break them down into other known functions, such that their linear combination minimizes the mean square error. Thus, this research proposes a wavelet linear combination of functions linked by a connectionist model, where the wavelet functions are generated by the cascade algorithm from orthogonal perfect-reconstruction filters, and by scaling and translating an objective function which is approximated. The proposed model has been called EPWavenets, as an abbreviation for the combination of evolutionary algorithms, filters parameterization, wavelet functions and artificial neural networks.

Thus, it seeks to demonstrate that it is possible to approximate a function of finite energy using wavelet functions generated with parametric filters whose combination is determined by the architecture of a neural network in where the parameters of scaling and translation, the synaptic weights and the parameters of the filters are determined with an evolutionary algorithm from the standpoint of supervised learning.

To demonstrate what was mentioned, the following methodology was used:

1. Review existing neural network models that are capable of modifying their basic functions and adjust their parameters to approximate functions
2. Propose a neural network model that allows including new basis functions and adjusting its parameters as monitored
3. Identify properties and families of wavelet functions that can be incorporated into the proposed model
4. Identify an evolutionary algorithm to adjust the parameters of the neural network model proposed
5. Identify a set of test functions to compare the results with other neural networks with different base functions
6. Generate comparative graphs and tables where can be appreciated the verification of the proposed hypothesis based on experimental results

The experimental results support the hypothesis indicating that it is possible to use wavelets functions with parametric filters to approximate finite energy functions in a neural network architecture.

It was also confirmed that it was possible to use evolutionary algorithms to optimize the free parameters of EPWavenets, in order to minimize the approximation error.

Derived from the results it is concluded that the EPWavenets achieve a high degree of adaptability and competitive performance compared to other neural networks involving radial basis functions on a set of test functions.

# Índice

## Índice de Figuras



## Índice de Tablas

**Tabla de nomenclaturas**

| | |
|---|---|
| **AG** | Algoritmo Genético |
| **AMR** | Análisis Multiresolución |
| **COMIA** | Congreso Mexicano de Inteligencia Artificial |
| **ECG** | Electrocardiograma |
| **FBR** | Función de Base Radial |
| **FORP** | Filtros Ortogonales de Reconstrucción Perfecta |
| **MLP** | Multi-Layer Perceptron |
| **RCS** | Research in Computing Science |
| **RNA** | Redes Neuronal Artificial |
| **RNFBR** | Red Neuronal de Base Radial |
| **RNFBRG** | Red Neuronal de Base Radial Gaussiana |
| **SLP** | Single Layer Perceptron |
| **TAU** | Teorema de Aproximación Universal |
| **VRA** | Visual Recurrence Analisys |

# Capítulo 1. Introducción

## 1.1 Antecedentes

Una de las motivaciones que dieron origen a las redes neuronales artificiales (RNA) fue la inspiración en el modelo conexionista del cerebro humano (Sotelo, 2002). Posteriormente, ante los cuestionamientos del cómo y por qué funcionaban las redes neuronales y en vista a hacer un mejor uso de ellas, se estableció la conexión con otras áreas del conocimiento, en particular con el Análisis Funcional que es una rama del Análisis Matemático orientada a estudiar las propiedades de funciones que generan espacios vectoriales. El Teorema de Aproximación Universal (TAU), por ejemplo, permite abordar desde esta perspectiva la capacidad de aproximar funciones de energía finita a partir de funciones no constantes, acotadas, monótonas crecientes y continuas (Cybenko, 1989). Así, en efecto, el TAU guarda una estrecha relación con las redes neuronales, en particular con las redes de perceptrones (Widrow, 1990).

Otro tipo de redes neuronales son las llamadas *redes neuronales de funciones de base radial* (Chen, 1991) cuya expresión matemática para aproximar una función $f(x)$ está dada por:

$$f(x) \approx \sum_{i=1}^{N} w_i \mu(||x - x_i||) + w_0 \qquad (1)$$

en donde $x \in R^n$ es el vector de entrada de datos en *n* dimensiones, $N$ es el número de unidades de procesamiento o neuronas, $x_i \in R^n$ son los centroides de las funciones de base radial, $w_i$ son los pesos sinápticos, $w_0$ es el umbral de desplazamiento, $||x - x_i||$ es la métrica de distancia, y $\mu(\cdot)$ es la función de base radial.

En trabajos previos se han propuesto varias funciones de base radial entre las que se encuentran las funciones: gaussiana, multicuadrática, multicuadrática inversa, y *splines* (Haykin, 1998). Otras base son las *wavelets*, con las que se dió origen a las redes neuronales *wavenets* (Delyon, 1995; Zhang, 1992).

Para aproximar una función $f(x)$ mediante funciones wavelets $\psi(x)$ se puede usar la expresión:

$$f(x) \approx \sum_{i=1}^{N} w_i \psi_{a_i,b_i}(x) = \sum_{i=1}^{N} w_i \psi(\frac{x - b_i}{a_i}) \qquad (2)$$

en donde los $a_i \neq 0$ son parámetros de escalamiento, y los $b_i \in R$ son parámetros de desplazamiento de la función wavelet principal $\psi(x)$.

El problema de entrenamiento de la red neuronal descrita en la ec. ( 2 ) radica en determinar el conjunto óptimo de parámetros libres que incluyen: el conjunto de valores $a_i$ y $b_i$, y los pesos sinápticos $w_i$ con $i = 1,2,\dots,N$, en donde $N$ es el número de neuronas.

Como se mencionó previamente relacionado con las redes neuronales está el análisis de funciones. Una manera de analizar una función consiste en hacer una descomposición en otras funciones conocidas que sean más fáciles de manipular, y de las cuales se tiene más información que la original. Las funciones conocidas suelen elegirse para que cumplan ciertas propiedades como la *ortogonalidad*. Ejemplo de funciones ortogonales son las funciones seno y coseno, usadas en las series de Fourier, y los polinomios de Legendre. La ortogonalidad entre dos funciones $f_1(x),\ f_2(x)$ se comprueba mediante la integral:

$$\int_{-\infty}^{\infty} f_1(x) f_2(x) dx = 0 \qquad (3)$$

El análisis y la aproximación de funciones permiten estudiar problemas más complejos. Por ejemplo, podemos obtener una aproximación de la integral de $e^x$ a través de su serie de Taylor descrita en la ec. ( 4 ):

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots + \frac{x^n}{n!} \qquad (4)$$

Y proceder a hacer la integración término a término para obtener la aproximación de la ec. ( 5 ):

$$\int e^x dx = x + \frac{x^2}{2} + \frac{x^3}{3 * 2!} + \frac{x^4}{4 * 3!} + \cdots + \frac{x^{n+1}}{(n+1)!} \qquad (5)$$

Otro tipo de funciones que recientemente ha tenido auge, desde el punto de vista del análisis matemático y de aplicaciones en ingeniería son las funciones wavelets. Las funciones wavelets toman su nombre del diminutivo de “onda”, es decir “ondita”, que alude a funciones que concentran su energía en un intervalo finito y después se desvanecen, lo cual las hace útiles para estudiar fenómenos transitorios, más que fenómenos estacionarios como sería al usar la transformada de Fourier, toda vez que las funciones sinusoidales (seno y coseno) oscilan en todo el eje real, y por lo tanto su energía se dispersa por todo el eje temporal.

Varios trabajos se han publicado sobre el cómo generar funciones wavelet (Kobayashi, 1994; Dominguez, 2012; Herrera, 2011), así como el estudio de sus propiedades. La mayoría de las funciones wavelets conocidas no tienen una expresión analítica (como las funciones seno y coseno) y sus propiedades están dadas por condiciones matemáticas, que dan lugar a familias de funciones bajo ciertos criterios de diseño. Algunos de esos criterios se enfocan en hacer que los wavelets concentren su energía en un intervalo lo más corto posible, o que sean funciones derivables, o que sean parametrizables.

La aproximación de funciones con wavelets desde el punto de vista conexionista (de redes neuronales) sigue siendo un área de investigación contemporánea, con relevancia científica, y con resultados promisorios con aplicaciones en otras áreas del conocimiento, como puede revisarse en las siguientes referencias:

- En este trabajo (Kobayashi, 1994) se usa una red neuronal con wavelets en un modelo de tres capas (1, N, 1), donde el proceso de aproximación se divide en dos partes. En la primera parte se ajustan los coeficientes de la wavelet y las neuronas, en tanto que en la segunda parte se utiliza un algoritmo de retropropagación para ajustar el error de aproximación.
- En este trabajo (Ning, 2008) se utiliza un modelo que utiliza redes neuronales wavelet y redes de funciones de base radial para realizar la aproximación.
- En este trabajo (Herrera, 2011) se hace uso de funciones wavelet paramétricas que se optimizan para comprimir al máximo una imagen dada, con la menor perdida de información.

## 1.2 Planteamiento del problema

La aproximación de funciones de energía finita se puede llevar a cabo al combinar funciones base (Cybenko, 1989). En investigaciones previas se han propuesto diferentes funciones base. Ejemplos de éstas son: Las series de Taylor (Briggs, 2014), las series de Fourier (Hsu, 2000), los polinomios de Legendre (Jackson, 1999), y la transformada wavelet (Daubechies, 1992; Soman, 2010). Cada una de esas funciones base tiene propiedades distintivas.

Esto ha dado lugar a las siguientes preguntas:

1. ¿Qué tipo de funciones base se pueden usar para aproximar funciones de energía finita?
2. ¿Cómo se pueden generar funciones base para aproximar funciones de energía finita?
3. ¿Es posible usar modelos de redes neuronales para aproximar funciones de energía finita?

Un acercamiento a las respuestas de estas preguntas, basado en trabajos de investigación previos, sería:

1. Sí es posible usar funciones ortogonales para aproximar funciones de energía finita (Cybenko, 1989). Sin embargo, de acuerdo al Teorema de Non-Free Lunch (Wolpert, 1997), no se puede esperar que exista una función ortogonal con la cual se puedan aproximar apropiadamente un conjunto suficientemente grande de funciones. En este trabajo de investigación se propone adaptar funciones base a diferentes funciones objetivo.
2. Sí es posible generar funciones ortogonales mediante la parametrización de filtros de reconstrucción perfecta (Roach, 2002). Al respecto, se hace notar

que aún no se ha encontrado evidencia de trabajos previos en donde se hayan incluido este tipo de parametrizaciones en redes neuronales para aproximar funciones.

3. Sí es posible usar modelos de red neuronal que al combinar funciones base aproximen funciones de energía finita.
   Como ya se mencionó previamente, existe al menos un teorema con un resultado contundente que respalda esta afirmación (Cybenko, 1989). Ejemplo de ello son las redes de base radial en donde se usan funciones gaussianas, y redes wavenets en donde se usan wavelets. Se hace notar que parte de presente trabajo de investigación consiste en ampliar el número de funciones wavelets involucradas en las aproximaciones.

Relacionado con las preguntas anteriores, se plantean las preguntas de esta investigación:

1. ¿Es posible determinar nuevos modelos de red neuronal que mejoren la aproximación de funciones de energía finita?
2. ¿Qué algoritmo puede aplicarse para ajustar los parámetros en un nuevo modelo de red neuronal?
3. ¿Es factible aplicar las tecnologías existentes para implementar un modelo de red neuronal con funciones paramétricas y ajustar sus parámetros con algoritmos evolutivos?

Para lo anterior se puede argumentar que:

1. Es posible proponer nuevos modelos de red neuronal modificando la arquitectura (conexiones entre neuronas) y usando funciones base generadas paramétricamente. Esta es una de las aportaciones del presente trabajo de investigación.
2. Es posible usar algoritmos de optimización evolutiva para ajustar los parámetros de una red neuronal para aproximar funciones. El argumento principal es que los algoritmos evolutivos son métodos generales de optimización que se adaptan a contextos particulares, en donde las técnicas de optimización local no ofrecen buenos resultados.
   Incluir el uso de algoritmos evolutivos, y en particular de algoritmos genéticos, para optimizar los parámetros de funciones base, y otros parámetros de la red neuronal, es una de las aportaciones del trabajo de investigación.
3. Es posible implementar nuevos modelos de redes neuronales con computadoras actuales en lenguajes de programación orientada a objetos, para optimizar los parámetros de las redes neuronales usando algoritmos evolutivos. Esta es una de las aportaciones del presente trabajo de investigación.

De esta forma se plantea lo siguiente:

Proponer un modelo de red neuronal que utilice funciones base paramétricas para aproximar funciones de energía finita en un intervalo de tiempo dado y cuyos parámetros sean ajustados con algoritmos evolutivos.

Aunque existen trabajos previos de aproximación de funciones con redes neuronales y funciones wavelets, es importante remarcar que la originalidad de este trabajo de investigación radica en proponer una arquitectura de red neuronal que use funciones wavelet generadas con filtros paramétricos, y que puedan usarse como aproximadores de funciones, en donde los parámetros se optimicen con algoritmos evolutivos.

## 1.3 Objetivos

### 1.3.1 Objetivo general

Proponer un modelo de red neuronal que aproxime funciones de energía finita usando funciones wavelets generadas con filtros paramétricos optimizados con un algoritmo evolutivo.

### 1.3.2 Objetivos específicos

- Proponer un modelo de red neuronal que utilice funciones wavelets generadas con filtros paramétricos de reconstrucción perfecta, para aproximar funciones de energía finita.
- Codificar el modelo de red neuronal en individuos de un algoritmo evolutivo para aproximar funciones discretas.
- Comparar el desempeño del modelo de red neuronal cuyos parámetros se optimizan con un algoritmo evolutivo con el desempeño de una red neuronal que usa funciones de base radial.

## 1.4 Delimitación

En este proyecto de investigación se trabajará con un modelo de red neuronal cuyas funciones base corresponden a funciones wavelet generadas a partir de filtros paramétricos de reconstrucción perfecta.

La construcción de funciones wavelet se fundamenta en criterios de diseño de filtros ortogonales paramétricos. En efecto, se pueden construir funciones wavelet a partir de filtros paramétricos de diferente longitud, y en esta investigación sólo se trabajará con filtros de longitud 4 y 6.

No existe una demostración formal para limitar la longitud del filtro a utilizar aunque, en la práctica, el tiempo de cómputo requerido para realizar las aproximaciones podría ser prohibitivo y al incrementar la longitud del filtro paramétrico no se obtendría un aporte significativo a la hipótesis de este trabajo, que es sobre la posibilidad de aproximar una función de energía finita a un nivel aceptable con funciones wavelets paramétricas.

Por lo tanto, teniendo en cuenta que el número de parámetros crece linealmente con la longitud del filtro, y para fines de delimitar el problema se ha decidido acotar la longitud de los filtros a 4 y 6 mismos que tienen 1 y 2 parámetros respectivamente (Roach, 2002).

Los criterios de diseño de filtros ortogonales establecen condiciones que al cumplirse garantizan la obtención de funciones wavelet ortogonales con las que se pueden aproximar funciones de energía finita (Hereford, 2003).

Una función $f(x)$ con $x \in R$ se dice que tiene energía finita si

$$\int_{-\infty}^{\infty} |f(x)|^2 dx < \infty \qquad (6)$$

Y en el caso discreto para $f(x)$, donde $x = x0, x0 + \Delta x, x0 + 2\Delta x$ ... se dice que $f(x)$ es de energía finita si

$$\sum_{x0}^{x0+n\Delta x} |f(x)|^2 < \infty, \text{ para n=0, 1, 2, …} \qquad (7)$$

Una representación visual de la función $f(x)$ descrita en ec. ( 7 ) se muestra en la Figura 1, con n=0, 1, 2 y 3.

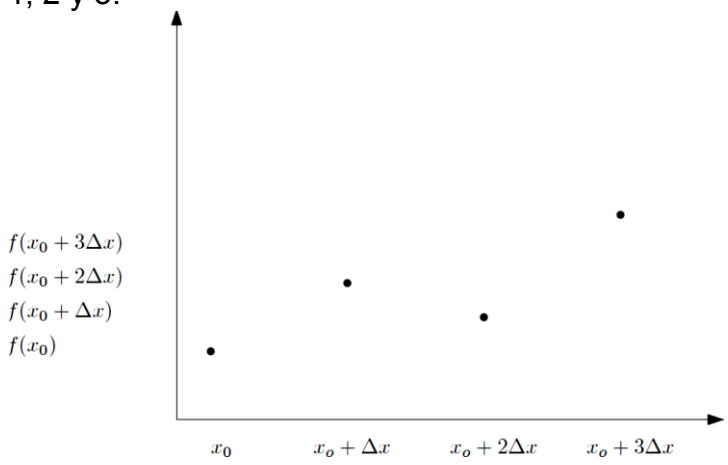



Figura 1. Representación de función discreta

En la presente investigación sólo se considerará el estudio del caso de funciones discretas con energía finita (ver ec.( 7 )).

## 1.5 Hipótesis

Dada una función de energía finita, es posible aproximarla con funciones wavelets generadas con filtros paramétricos cuya combinación está determinada por la arquitectura de una red neuronal en donde los parámetros libres se determinan con un algoritmo evolutivo.

## 1.6 Justificación

La aproximación de funciones por descomposición en componentes fundamentales tiene varias justificaciones y aplicaciones en diversas áreas del conocimiento, entre las cuales se pueden mencionar la mecánica, el análisis funcional, y el análisis de circuitos eléctricos.

En la mecánica, como subrama de la física, la descomposición de fuerzas F en componentes horizontales Fx y verticales Fy permite el cálculo de momentos (Halliday, 2001) y la solución de múltiples problemas con aplicaciones del mundo real.

En el análisis funcional se plantea que una función de energía finita f puede descomponerse como combinaciones lineales de funciones base de un espacio vectorial V (o espacio de funciones) (Folland, 1984). También existen aplicaciones de esto, como es en la solución de ecuaciones diferenciales.

En ingeniería eléctrica, el análisis de circuitos eléctricos se puede realizar bajo principios de superposición, analizando en forma separada los circuitos eléctricos con funciones base que conforman a una señal compuesta (Dorf, 2006).

Las series de Fourier (Hsu, 2000) también son un ejemplo de descomposición de funciones, en donde las funciones base corresponden a senos y cosenos de diferente amplitud ($w_0$) y frecuencia ($f_0$), como se puede ver en la ec. ( 8 ):

$$f(x) \approx w_0 \sin(2\pi f_0 t) + w_0 \sin(4\pi f_0 t) + w_0 \sin(6\pi f_0 t) + \cdots + w_0 \sin(n\pi f_0 t) \qquad (8)$$

Un área del conocimiento relacionada con el análisis funcional son las redes neuronales, en donde la idea fundamental es que una combinación lineal de funciones base permite aproximar funciones de energía finita $f(x)$.

Las funciones $f(x)$ podrían tener una expresión analítica, pero también se puede dar el caso en donde sea desconocida, como es la función del precio del dólar. Las gráficas del precio del dólar, que frecuentemente se muestran en los periódicos, en la televisión, y otros medios de comunicación, se construyen a partir de muestras sobre un eje temporal, como se describe en la Figura 1.

Las redes neuronales se han usado para aproximar funciones con expresiones analíticas conocidas y desconocidas. En este contexto, la aproximación de

funciones adquiere gran importancia ya que entre otras aplicaciones facilitan la predicción. Por ejemplo, en la predicción del clima (Maqsood, 2004).

La justificación de la aproximación de funciones de energía finita a partir de wavelets está soportada matemáticamente por la convergencia de series. La convergencia de una sucesión de funciones $f1, f2, f3,\ldots, fn$ hacia una función $f(x)$ de energía finita se expresa así:

$$\lim_{n\to\infty} |fn - f(x)| < \epsilon \qquad (9)$$

Donde $\epsilon > 0$.

En el caso particular de funciones wavelet, una expresión alternativa de la ec. ( 9 ) es la fórmula de reconstrucción para $f(x)$:

$$f(x) \approx w_1\psi\left(\frac{x-b_1}{a_1}\right) + w_2\psi\left(\frac{x-b_2}{a_2}\right) + \cdots w_n\psi\left(\frac{x-b_n}{a_n}\right) \qquad (10)$$

En donde:

$\psi$ es la función wavelet principal (base, o fundamental)
$a_i$ es un coeficiente de escalamiento
$b_i$ es un coeficiente de traslación (desplazamiento o localización temporal)
$w_i$es un coeficiente de transformación wavelet
$n$ es el número de términos de aproximación

Inspirados en ec. ( 10 ) es posible modelar una red neuronal para hacer una aproximación de funciones a partir de funciones wavelets $\psi$, en donde el objetivo es combinar el menor número de neuronas (términos de aproximación, n) y la selección apropiada de los valores de escalamiento, de traslación y de coeficientes de transformación.

## 1.7 Fundamentación inicial

Las wavelets son funciones que tienen propiedades que las hacen una herramienta poderosa en la teoría de la aproximación. Tales propiedades como son el soporte compacto, la ortogonalidad, y la localización en tiempo y frecuencia han ocasionado que el uso de wavelets sea abordado de diferentes maneras para la aproximación de funciones desde hace varios años (Soman, 2010).

Las wavenets son un modelo de red neuronal que utiliza wavelets como funciones base. Las wavenets son utilizadas como un enfoque alterno al uso de redes neuronales tradicionales, como son las redes de base radial. Uno de los primeros en desarrollar wavenets fue Benveniste (Zhang, 1992), quien propuso una red neuronal de capa $(1+\frac{1}{2})$ con wavelets como funciones base, además de usar un algoritmo de retropropagación para entrenar dicha red.

En (Kobayashi, 1994) se hacen uso de un algoritmo redes neuronales con wavelets. Ese algoritmo está dividido en dos partes, en la primera parte se utiliza un modelo de red de tres capas (1, N, 1) donde la función base es una función wavelet principal, y las capas de entrada y salida son funciones lineales. En esta etapa los coeficientes de escalamiento, de dilatación, el número de neuronas y los pesos se calculan simultáneamente usando la regla de *Kohonen*. En la segunda parte se utiliza un *algoritmo de retropropagación* para ajustar el error de aproximación.

Otro trabajo de aproximación de funciones es (Bakshi, 1992) en el que utiliza un modelo de red de base radial con una función wavelet base y un algoritmo de gradiente para ajustar los pesos de la red.

En trabajos recientes, como (Ning, 2008) y (Dominguez, 2012), se reportan estudios comparativos entre las redes neuronales de bases radial y la redes neuronales con wavelets, y se demuestra que el uso de estas últimas mejora los resultados de aproximación en diferentes aspectos como son el error de aproximación, el tiempo de entrenamiento y el número de neuronas utilizadas.

Observese que la mayoría de los trabajos anteriores se han enfocado en el uso de wavelets para la aproximación de funciones utilizando métodos de gradiente para ajustar los parámetros de dilatación y traslación y así minimizar el error de aproximación, lo que implica conocer la función a aproximar y de las funciones wavelets en su forma analítica.

En el trabajo actual se pretende proponer un modelo de red neuronal que funcione de manera similar a las redes neuronales de base radial, pero incorpore el uso de algoritmos evolutivos para ajustar los parámetros de dilatación y traslación de funciones wavelets, así como los pesos sinápticos de las neuronas y los parámetros de los filtros digitales que generan las funciones wavelet para mejorar la aproximación reduciendo el error cuadrático medio.

En trabajos previos se han propuesto varias funciones wavelets $\psi$ con diferentes propiedades, sin embargo solo se han identificado un número finito de ellas que cuentan con expresión analítica.

Es importante mencionar que en este trabajo de investigación no se determinará la expresión analítica de las funciones wavelets generadas a partir de los filtros de reconstrucción perfecta sino que, alternativamente, es posible aproximarlas a un nivel aceptable (Mallat, 1989).

En efecto, en esta investigación se propone generar un gran número de funciones wavelets a partir de filtros paramétricos de reconstrucción perfecta, por lo que adicional a los parámetros de la red neuronal también se incluyen los parámetros determinados por los criterios de diseño de filtros de longitud 4 y 6.

La parametrización de filtros (y consecuentemente de funciones wavelets) garantiza que todas las funciones generadas tengan soporte compacto

(determinado por la longitud del filtro) y sean una base del espacio vectorial de funciones de energía finita, con lo cual se reduce la redundancia y el número de neuronas requeridas para una aproximación.

## 1.8 Metodología

Desarrollo. Para cumplir con los objetivos en esta investigación y probar la hipótesis planteada se han identificado las siguientes etapas:

1. Modelo de red neuronal
2. Funciones wavelets y filtros paramétricos
3. Algoritmos evolutivos y redes neuronales
4. Universo de funciones de prueba
5. Experimentos
6. Conclusiones

Para cumplir con los objetivos en esta investigación y probar la hipótesis planteada se propone seguir la siguiente metodología general:

Etapa 1. Revisar modelos existentes de redes neuronales y proponer una variante
Etapa 2. Identificar funciones base, propiedades y aplicaciones en redes neuronales
Etapa 3. Implementar el modelo de red neuronal propuesto
Etapa 4. Validar la hipótesis con pruebas experimentales

En estas etapas se realizaron las siguientes actividades:

- Revisar modelos de red neuronal existentes que sean susceptibles de modificar sus funciones base y ajustar sus parámetros para aproximar funciones
- Proponer un modelo de red neuronal que permita incluir nuevas funciones base y ajustas sus parámetros en forma supervisada
- Identificar propiedades y familias de funciones wavelet que puedan incorporarse al modelo propuesto
- Identificar un algoritmo evolutivo para ajustar los parámetros del modelo de red neuronal propuesto
- Identificar un conjunto de funciones de prueba para comparar los resultados con otras redes neuronales con diferentes funciones base
- Generar tablas comparativas y gráficas en donde se aprecie la comprobación de la hipótesis planteada a partir de los resultados experimentales

**Variables de estudio**

Se identifican las siguientes variables:

- El modelo de la red neuronal
- Las funciones base

Cabe mencionar que aunque se propone utilizar algún método de optimización evolutiva, este no se considera como variable ya que no se propone aportar alguna variante o mejora.

**Universo de funciones de prueba**

Para poner en práctica el modelo propuesto se identificaron 13 funciones $f(x)$ de energía finita cuyos valores se normalicen en el intervalo [0, 1], estas funciones ser tomaron en el Benchmark Visual Recurrence Analisys (VRA) (Kononov, 2010).

Cada función $f(x)$ está determinada por valores discretos de $x$, cómo se describió en la Sección 1.4, como se ejemplifica en la Figura 1.

En los experimentos se realizarán la aproximación de funciones wavelets a partir de parametrizaciones de filtros de longitud L = 4 y L = 6, los cuales se describirán en el Capítulo 2.

Para determinar el mejor modelo para cada función correspondiente se pretende variar el número de neuronas de 1 hasta 10, esto con cada tamaño de filtro L=4 y L=6. El criterio para elegir el mejor modelo será el error cuadrático medio el cual tiene la siguiente expresión:

$$Error = \frac{1}{M}\sum_{m=1}^{m=M}\sum_{i=1}^{n}\left(x_i(m) - y_i(m)\right)^2 \qquad (11)$$

en donde *M* es el número de muestras de los datos de entrada, $x_i(m)$ es la *i*-ésima componente del dato de entrada en el instante *m*, y $y_i(m)$ es la *i*-ésima componente de la salida de la red en el instante *m*.

**Experimentos**

Los experimentos a realizar siguen los siguientes pasos:

1. Considerar N funciones de prueba $\{f(x)\}_N$.
2. Considerar cada una de las funciones objetivo $f(x)$ tomadas de $\{f(x)\}_N$.
3. Identificar la mejor arquitectura de aproximación experimental de una red neuronal con funciones wavelet optimizadas evolutivamente que minimiza el error cuadrático medio sobre las M muestras de una función $f(x)$.
4. Generar una red neuronal con la arquitectura obtenida en el paso 3 cambiando las funciones wavelets por funciones gaussianas.

5. Optimizar los parámetros de la red generada en el paso 4 y aplicar el algoritmo evolutivo para minimizar el error cuadrático medio sobre las M muestras de una función $f(x)$.
6. Comparar los resultados de aproximación de los pasos 3 y 5.

## 1.9 Publicaciones derivadas de este trabajo

Se presentó una ponencia titulada “Aproximación con EPWavenets” en el Congreso Mexicano de Inteligencia Artificial COMIA 2015 organizado por la Sociedad Mexicana de Inteligencia Artificial, en donde además se obtuvo el Premio al Segundo Lugar en la categoría de mejor artículo de investigación sometido en 2015. El artículo fue publicado en un número especial de la revista Research in Computing Science (RCS) con ISSN, indizada en Latinindex y DBLP (Romero, 2015).

## 1.10 Organización del capitulado

La organización de la tesis en 6 capítulos. En el Capítulo 1 se presentó la introducción al trabajo de tesis, con el planteamiento del problema, los objetivos tanto generales como específicos, la hipótesis que sustenta la investigación, la justificación de la investigación, la fundamentación inicial y la metodología.

En el Capítulo 2, se presenta el Marco Teórico sobre los temas de redes neuronales de funciones de base radial, funciones wavelets y su relación con filtros ortogonales paramétricos de reconstrucción perfecta, así como de algoritmos evolutivos. Después de haber presentado los conceptos fundamentales sobre redes neuronales, wavelets y algoritmos evolutivos, en el Capítulo 3 se presentan las *EPWavenets* como un modelo de red neuronal que introduce funciones wavelets paramétricas, susceptibles de ser adaptadas para aproximar funciones de energía finita mediante combinaciones lineales.

También en el Capítulo 3 se describe el desarrollo de la tesis, en donde se presenta el cómo se realiza la aproximación de funciones discretas, el cómo se configura el algoritmo evolutivo para aproximar funciones de energía finita, y se presentan las funciones de prueba utilizadas.

En el Capítulo 4, se describen los experimentos realizados y sus correspondientes resultados. Un primer experimento consiste en encontrar la mejor EPWavenet experimental para cada una de las funciones de prueba. Un segundo experimento consiste en comparar los resultados de aproximación de EPWavenets del primer experimento, con redes neuronales que usan funciones gaussianas. Y un tercer experimento que permite ver el desempeño como predictores de funciones de energía finita a las EPWavenets y a las redes neuronales con funciones gaussianas y sigmoidales.

En el Capítulo 5 se presentan las conclusiones obtenidas y trabajos futuros relacionados con la presente investigación, en base a los experimentos y resultados del Capítulo 4.

# Capítulo 2. Marco teórico

## 2.1 Redes neuronales

En los años 50 Warren S. McCulloch y Walter Pitts (McCulloch, 1943) realizaron estudios sobre el sistema neuronal biológico, y propusieron uno de los primeros modelos de una neurona. Una neurona tiene las características de procesar información en forma paralela y no lineal, lo cual la hace una herramienta útil en el tratamiento de problemas que son no lineales por naturaleza. Para el ámbito computacional una neurona artificial se puede definir como una unidad de procesamiento que dado un conjunto o vector de entrada produce una salida única.

Una neurona artificial tiene las siguientes características:

- Está compuesto por conjunto de entradas $x_j(t)$, los cuales son valores que se transmiten a la capa intermedia.
- Pesos sinápticos $w_{ij}$ los cuales son valores reales que conectan la entrada $j$ con la neurona $i$.
- Se tiene una función de transferencia la cual permite propagar la salida a otras neuronas, teniendo la siguiente forma.

$$h_i(t) = \sigma\left(w_{ij}, x_j(t)\right) \qquad (12)$$

La regla más común es $\sum w_{ij}x_i$ por lo que la ec. ( 12 ) puede expresarse de la siguiente forma:

$$h_i(t) = \sum w_{ij}x_j \qquad (13)$$

Donde $i$ es el número de neuronas $y$ $j$ es el número de entradas.

- Así, la función de activación que proporciona la salida actual de la neurona.

$$y_i(t) = f_i(h_i(t)) \qquad (14)$$

En la Figura 2 se muestra la estructura de una neurona artificial.

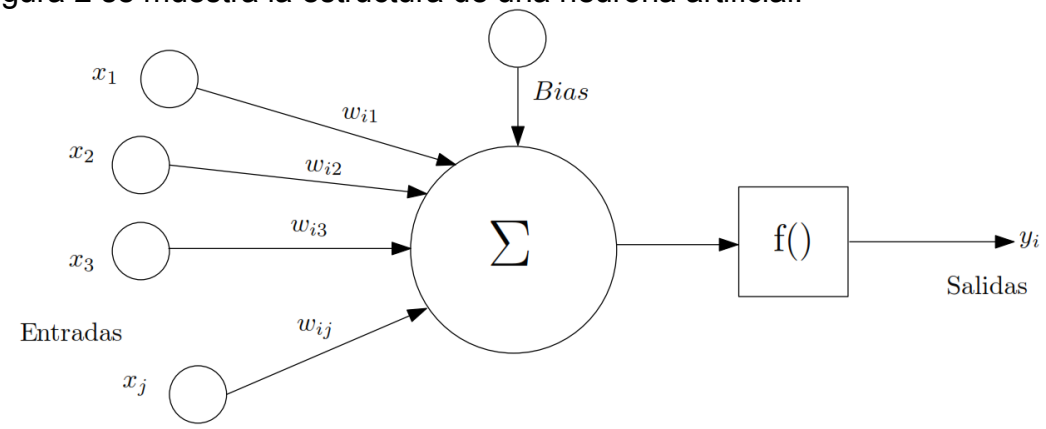



Figura 2. Estructura de una neurona artificial

En la Figura 3 (a, b, c y d) se muestran algunas de las funciones de activación más usuales:

- En la Figura 3a se muestra la función gaussiana, cuya expresión analítica es:

$$f(r) = e^{-r^2/2\sigma^2} \qquad (15)$$

En donde $r$ es el radio de la función gaussiana centrado en cero, y con desviación estándar $\sigma$.

- En la Figura 3b se muestra la función escalón, cuya expresión analítica es:

$$f(x) = \begin{cases} 1, & x \geq +\frac{1}{2} \\ x, & +\frac{1}{2} > x \geq -\frac{1}{2} \\ 0, & x \leq -\frac{1}{2} \end{cases} \qquad (16)$$

- En la Figura 3c se muestra la función sigmoidal, cuya expresión analítica es:

$$f(x) = \frac{1}{1 - e^{(-ax)}} \qquad (17)$$

  Donde $a$ es el parámetro de escalamiento.

- En la Figura 3d se muestra la función escalón, cuya expresión analítica es:

$$f(x) = \begin{cases} 1 & si \quad x \geq 0 \\ 0 & si \quad x < 0 \end{cases} \qquad (18)$$

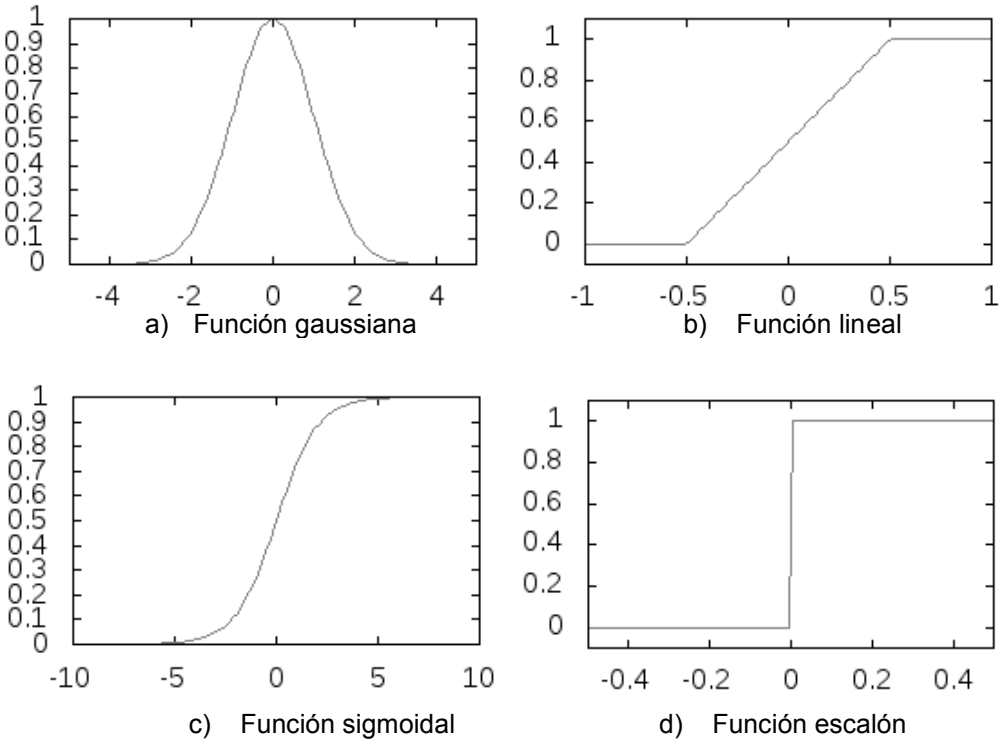


Figura 3. Funciones de activación

Generalmente al conjunto de pesos se le agrega un parámetro adicional $\theta$ llamado umbral o “bias”, el cual es un parámetro que le añade un grado de libertad adicional a la neurona, de esta manera la función de activación se expresa como en la ec. ( 19 ):

$$y = f\left(\sum w_{ij}x_j - \theta_i\right) \qquad (19)$$

Una red neuronal es un conjunto de nodos, que determinan el comportamiento de la red. Estos nodos se agrupan en capas las cuales a su vez se agrupan en conjuntos llamados *grupos neuronales*. Por último un conjunto de uno o más capas conforman la llamada red neuronal.

Una red neuronal típicamente constar de tres capas, la primera capa es la de entrada, donde las neuronas reciben los datos a analizar, la segunda es la capa oculta, donde las neuronas no tienen contacto con directo con el ambiente, y solo mantiene contacto con los más capas o grupos neuronales. Y por último, la tercera capa de salida donde las neuronas dan una respuesta de la red neuronal.

Pueden incluirse más neuronas entre la segunda capa (oculta) y la última capa de salida, para reducir el número de neuronas requeridas en cada capa oculta. Esto es consistente con el TAU que afirma que una sola capa es suficiente, pero no precisa el número de neuronas requeridas.

Se pueden identificar principalmente tres tipos de arquitecturas de red neuronal: perceptrones monocapa, perceptrones en multicapa y redes recurrentes. A continuación se da una breve descripción de cada una de ellas.

**Perceptrones monocapa (Single Layer Perceptron SLP)**

Esta arquitectura tiene una sola capa de entrada donde los nodos (perceptrones individuales) introducen los datos a la capa de salida, la información en esta arquitectura solo fluye en una dirección y es llamada "*feedforward*" o "acíclica". El nombre de monocapa hace referencia a que solo es tomada en cuenta la capa de salida, donde las neuronas realizan las operaciones (Ver Figura 4).

**Perceptrones en Multicapa (Multilayer Perceptron MLP)**

Esta arquitectura de red neuronal está formada por una o más capas ocultas. Esta capa oculta, como se mencionó anteriormente, hacen una conexión entre la entrada y la salida de la red neuronal. La información que llega en la capa de entrada genera un patrón de activación el cual es una señal de entrada aplicada a las neuronas de la capa oculta. Si existe más de una capa oculta la señal de salida de la primera capa oculta será la entrada de la segunda capa oculta y así sucesivamente hasta llegar a la capa de salida (Ver Figura 5).

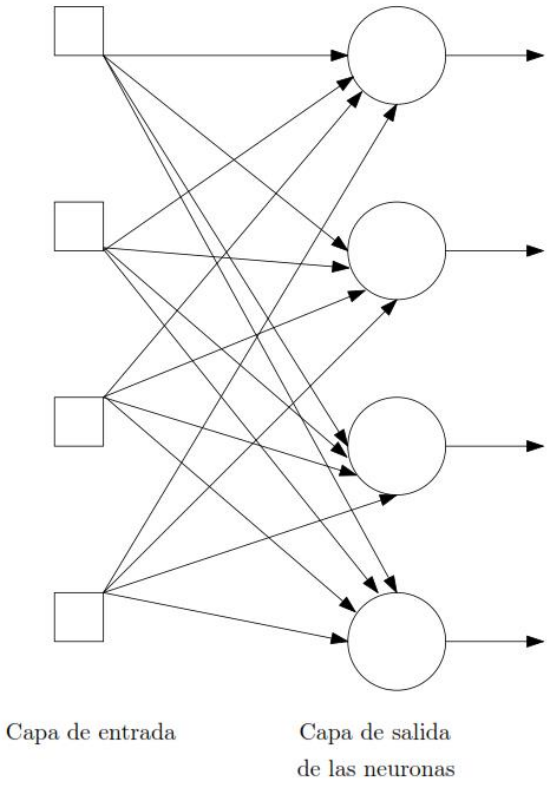



Figura 4. Arquitectura de una red neuronal SLP

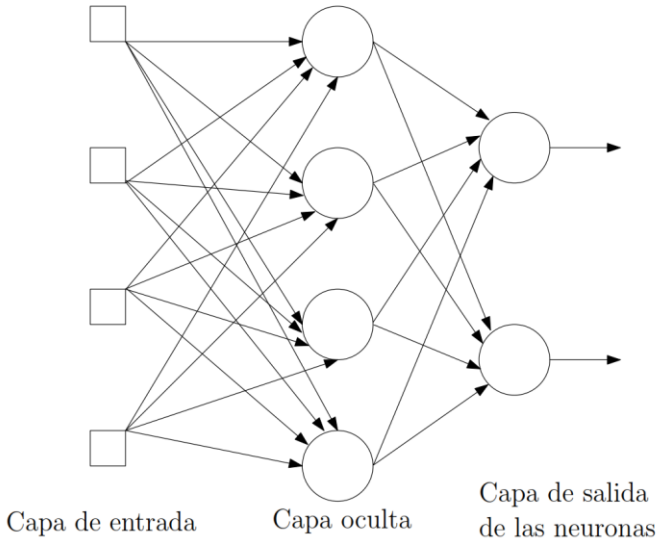



Figura 5. Arquitectura de una red neuronal MLP

**Redes recurrentes**

Una red neuronal recurrente contiene al menos una conexión de retroalimentación, esto provoca que las funciones de activación se ejecuten en un ciclo repetitivo. La arquitectura más común de una red neuronal recurrente es parecida a una MLP que le brinda la capacidad de tener memoria, además de mejorar la capacidad de hacer representaciones de espacios no lineales, (ver Figura 6). Otra característica es la incorporación de un elemento llamado unidad de retraso representado por $z^{-1}$ (Haykin, 1998).

## 2.2 Redes neuronales de base radial

Otro tipo de red neuronal son las Redes Neuronales de funciones de Bases Radiales (RNFBR) que utiliza un enfoque llamado “el problema de ajuste de curva en un espacio de alta dimensión”, que consiste en crear un dominio con un espacio multidimensional que genere el mejor ajuste de la red. (Powell, 1987).

La arquitectura de las Redes Neuronales de Base Radial define tres capas: la capa de entrada, la capa oculta, y la capa de salida. La capa oculta aplica una transformación no lineal a los datos provenientes de la capa de entrada para llevarlos a otro espacio vectorial que suele tener una mayor dimensión. Al pasar de un espacio a otro de mayor dimensión se explota de mejor manera la capacidad de las redes neuronales en tareas de clasificación y de aproximación de funciones (Haykin, 1998). Esta última tarea es de interés en el presente artículo.

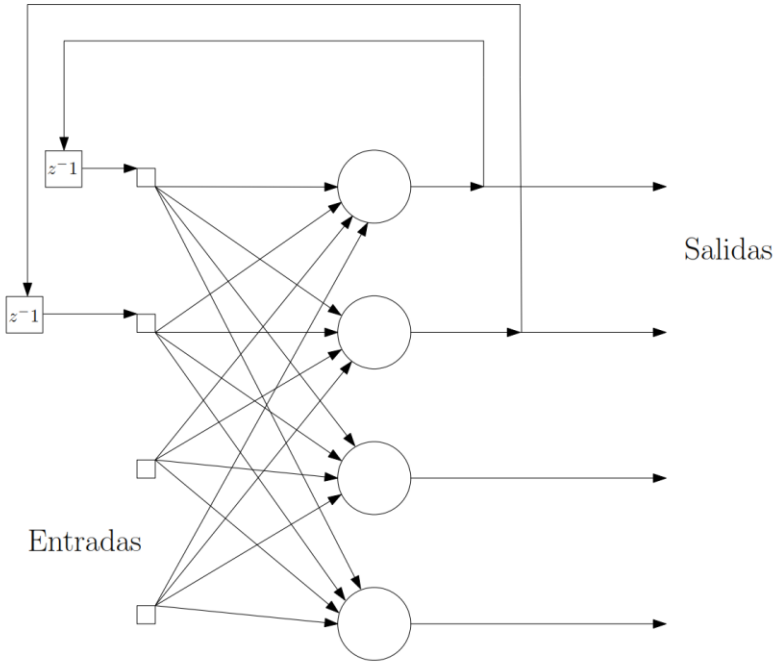



Figura 6. Arquitectura de una red recurrente

Una función de base radial tiene la característica de que la respuesta que genera disminuye o aumenta de forma gradual con la distancia a un punto fijo llamado centroide. Una propiedad de las RNFBR es que cualquier función de energía finita puede ser aproximada utilizando una combinación lineal de funciones de base radial.

Una RNFBR como aproximador de funciones descompone cada uno de los datos de entrada en varias componentes que pasan por $N$ neuronas, mismas que se suman en la capa de salida. La Figura 7 ilustra la arquitectura de una RNFBR acorde con la ec. ( 1 ) en donde cada muestra de entrada corresponde a un vector $x \in R^n$. Cada neurona incluye una función de base radial (FBR), entre las que se encuentran: la función gaussiana, la función cuadrática, la función cuadrática inversa, la función multicuadrática, y las funciones *splines*. La función gaussiana es ampliamente utilizada y su expresión esta dada.

$$f(r) = e^{-(r-\mu)^2/2\sigma^2} \qquad (20)$$

En donde $r$ es el radio respecto al centroide $\mu$, y $\sigma$ es un factor de escala que mide la anchura de la función gaussiana.

La capa de salida realiza una combinación lineal de las activaciones de las neuronas de la capa oculta considerando los pesos sinápticos $w_i$. El peso sináptico $w_0$ corresponde al valor de umbral (*BIAS*=1) que modifica el valor de la salida $Y$.

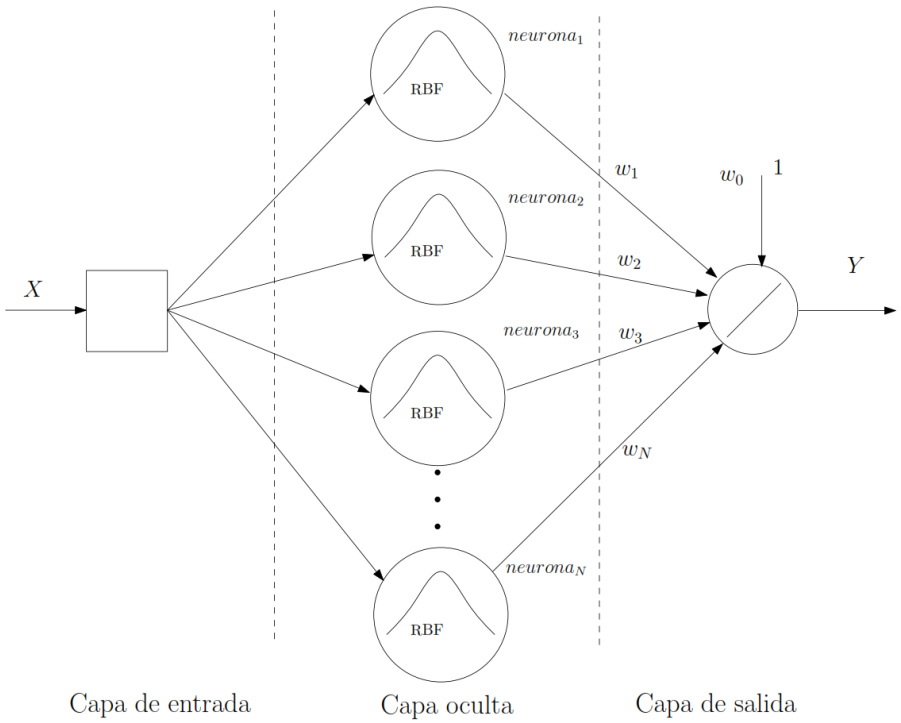



Figura 7: Una red neuronal de función de base radial

## 2.3 Wavelets

Las wavelets son familias de funciones $\psi_{a,b}(x) = a^{\frac{1}{2}}\psi(\frac{x-b}{a})$ con $a \neq 0$, que surgen de una función principal $\psi(x)$ que cumple las condiciones de ortogonalidad, y ortonormalidad $\int_{-\infty}^{\infty} |\psi(x)|^2 dx = 1$. Las funciones wavelets además cumplen la condición de admisibilidad dada por $\int_{-\infty}^{\infty} \frac{|\widehat{\psi}(w)|^2}{|w|} dw < \infty$ lo cual les da la propiedad de localización, que significa que su transformada de Fourier $\widehat{\psi}(w)$ se desvanece rápidamente. También se pueden generar wavelets con soporte compacto cuyo valor es cero fuera del intervalo de soporte en el dominio temporal.

Los primeros usos de las funciones wavelets fueron en el área de geología, con el fin de encontrar una mejor manera de localizar petróleo utilizando señales sismológicas. El uso de wavelets solucionó los problemas que se obtenían al usar análisis de Fourier, ya que existían cambios abruptos en la información que debía ser analizada. Jean Morlet, un ingeniero de Elf-Aquitiane, desarrolló su propio wavelet para analizar señales sismológicas para crear componentes que fueran localizados en el espacio de búsqueda (Büssow, 2007) (ver Figura 8).

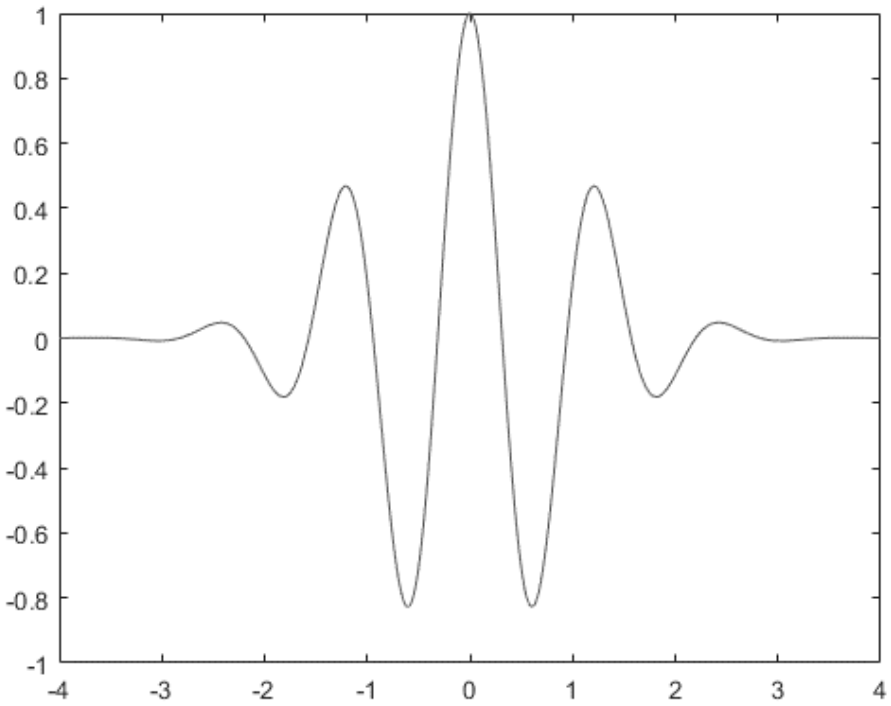



Figura 8. Wavelet de Morlet *(The MathWorks, 1994)*

Tiempo después el investigador Yves Meyer, investigando la teoría de las funciones Wavelets, desarrolló funciones wavelets con la propiedad matemática llamada ortogonalidad (Ver Figura 9), la cual permite que la información capturada por una wavelet sea completamente independiente por la información capturada por otra (Xudong, 2009). Esta propiedad hace que la transformada wavelet sea tan fácil de manipular como la transformada de Fourier.

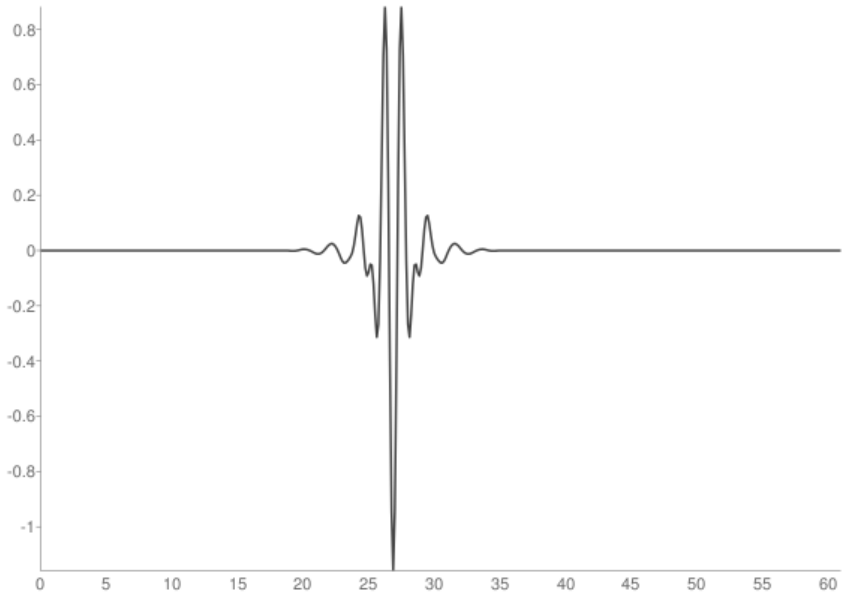



Figura 9. Wavelet de Meyer *(The MathWorks, 1994)*

Otro investigador que realizó un gran aporte a la teoría Wavelets fue Stéphane Mallat (Mallat, 1989), quien unió la idea del análisis multiresolución, que es observar las señales en diferentes escalas de resolución, con la generación de funciones wavelets. Esto permite realizar análisis wavelet sin la necesidad de conocer la fórmula de la wavelet principal.

Ingrid Daubechies (Daubechies, 1992), una profesora del departamento de matemáticas de la universidad de Princeton, propuso una nueva familia de wavelets que cumplían con todas la propiedades de las wavelets antes desarrolladas, pero que además tenían la característica de poder ser implementados con filtros digitales, lo que hizo que su programación fuera más fácil en computadoras digitales. En la Figura 10 se muestran algunos ejemplos de los wavelets Daubechies

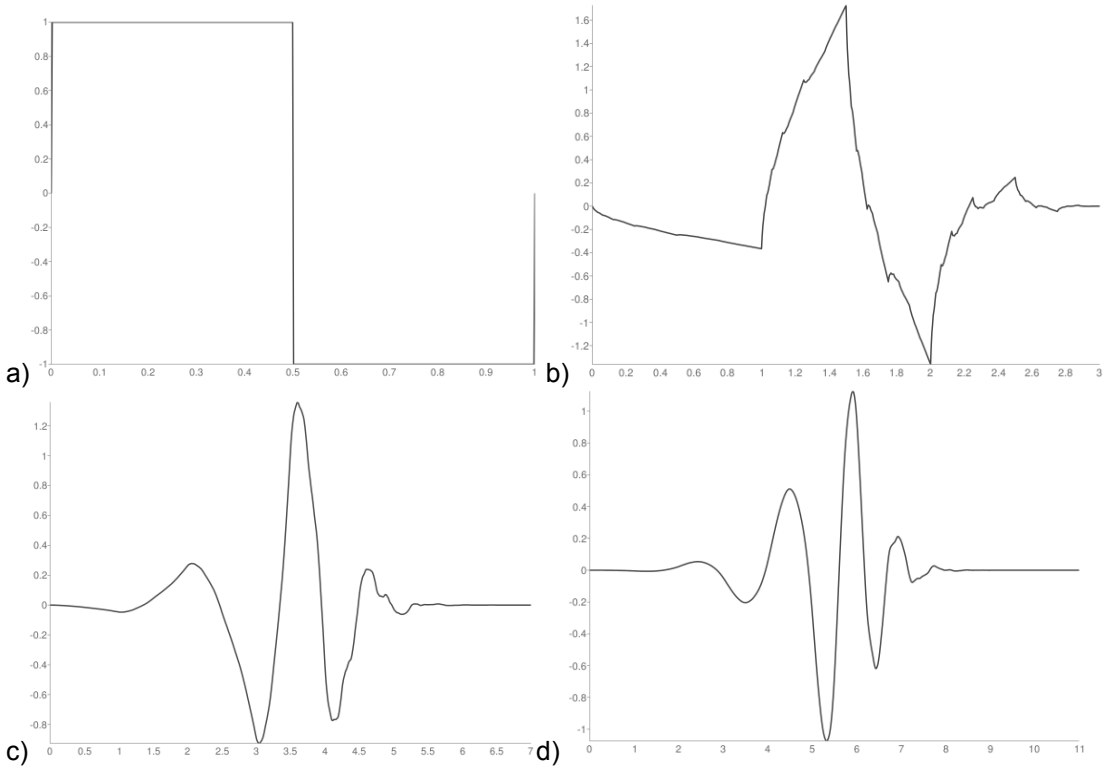



Figura 10. a) wavelet daubechies 1 (db1 o Haar) b) wavelet daubechies 2 (db2) c) wavelet daubechies 4 (db4) d) wavelet daubechies 6 (db6) *(The MathWorks, 1994)*

Cabe mencionar que durante el paso de los años se han generado más familias de wavelets como son los *Symlets*, los *Coiflets* y los *wavelets Biortogonales*, los cuales tienen características y propiedades similares a los wavelets Daubechies (Soman, 2010).

Existen diferencias importantes entre el uso de la transformda Fourier y de la *transformada Wavelet*. Las funciones base de la transformada Fourier sirven para la localización en frecuencia pero no en tiempo, y pequeños cambios en la transformada de Fourier podrían producir cambios en todo el dominio del tiempo.

Las wavelets por otra parte tienen localización en frecuencia (usando dilatación) y en tiempo (usando translación), lo cual les permite hacer una análisis tiempo frecuencia, por lo que la hace una herramienta para analizar fenómenos transitorios.

Otra ventaja es que muchas funciones pueden ser representadas de una forma más compacta usando wavelets. Por ejemplo, funciones discontinuas y funciones con puntas afiladas o picos, requieren el uso de menos funciones bases wavelets que funciones base seno y coseno para lograr una aproximación comparable.

La complejidad computacional de la transformada rápida de Fourier es de $O(n \log n)$ mientras que la complejidad de la transformada wavelet es $O(n)$.

Una forma de generar funciones wavelets es a través del análisis multiresolución (AMR) (Mallat, 1989) en donde se hace uso de funciones de escalamiento $\phi(x)$ ortogonales a las funciones wavelets $\psi(x)$. Para poder generar las wavelets debemos de tener un espacio vectorial $L^2(R)$ donde todas las funciones sean integrables, es decir, es el espacio de todas las funciones $f(x)$ para las cuales $\int_{-\infty}^{\infty} |f(x)|^2 dx < \infty$.

## 2.4 Filtros ortogonales de reconstrucción perfecta

De acuerdo con el AMR es posible generar wavelets ortogonales con soporte compacto a través de la parametrización de los coeficientes de un banco de filtros ortogonales $h$ y $g$ de reconstrucción perfecta (FORP). Los FORP constan de un filtro pasabajas $h$ y un filtro pasaaltas $g$. Un filtro $h = [h_0, h_1, h_2, h_3]$ de longitud $L = 4$, por ejemplo, genera wavelets con soporte compacto en el intervalo $[0,1]$, y sus coeficientes dependen de un parámetro $\alpha$ de la siguiente manera (Roach, 2002):

$$h_0 = \frac{1}{4} + \frac{1}{2\sqrt{2}}\, cos(\alpha) \qquad h_1 = \frac{1}{4} + \frac{1}{2\sqrt{2}}\, sen\,(\alpha)$$
$$h_2 = \frac{1}{4} - \frac{1}{2\sqrt{2}}\, cos(\alpha) \qquad h_3 = \frac{1}{4} - \frac{1}{2\sqrt{2}}\, sen(\alpha) \qquad (21)$$

en donde $\alpha \in [0,2\pi)$. Los correspondientes coeficientes del filtro pasaaltas se calculan a partir de los coeficientes $h_i$ mediante la ecuación $g_i = (-1)^i h_{N-i-1}$. En el caso de filtros de longitud $L = 6$ los coeficientes dependen de dos parámetros $\alpha$ y $\beta$ de la siguiente manera (Roach, 2002):

$$h_0 = \frac{1}{8} + \frac{1}{4\sqrt{2}} cos(\alpha) + \frac{p}{2}\cos(\beta), \qquad h_1 = \frac{1}{8} + \frac{1}{4\sqrt{2}} sen(\alpha) + \frac{p}{2} sen\,(\beta)$$
$$h_2 = \frac{1}{4} - \frac{1}{2\sqrt{2}} cos(\alpha), \qquad h_3 = \frac{1}{4} - \frac{1}{2\sqrt{2}} sen(\alpha) \qquad (22)$$
$$h_4 = \frac{1}{8} + \frac{1}{4\sqrt{2}} cos(\alpha) - \frac{p}{2} cos\beta, \qquad h_5 = \frac{1}{8} + \frac{1}{4\sqrt{2}} sen(\alpha) - \frac{p}{2} sen(\beta)$$

en donde $p = \frac{1}{2}\sqrt{1 + sen(\alpha + \frac{\pi}{4})}$, $\alpha, \beta \in [0, 2\pi)$.

Se puede demostrar que los FORP deben tener una longitud $L$ par, y que el soporte compacto del wavelet es $[0, \frac{L}{2} - 1]$.

Cabe mencionar que existen pocas funciones wavelet con soporte compacto que tengan una expresión analítica. Uno de los casos en los que sí se tiene la expresión analítica es para la función wavelet Haar descrita como:

$$f(x) = \begin{cases} 1 \quad si \quad x \in [0, \frac{1}{2}) \\ -1 \quad si \quad x \in [\frac{1}{2}, 1) \\ 0 \qquad caso\ contrario \end{cases} \tag{23}$$

Para los propósitos de esta investigación no será necesaria la expresión analítica de las funciones wavelets, toda vez que se aproximaran múltiples funciones mediante filtros digitales paramétricos.

## 2.5 Algoritmos evolutivos

Los algoritmos evolutivos tienen su inspiración en la selección natural y supervivencia del individuo mejor adaptado. Estos algoritmos difieren de otras técnicas ya que generan una población de soluciones. Mediante el uso de combinaciones y mutaciones se obtienen nuevos individuos que pueden generar mejores soluciones durante un número determinado de iteraciones. Existen diferentes técnicas o métodos en el área de algoritmos evolutivos, como son: Programación Genética, Programación Evolutiva y Algoritmos Genéticos, esta última es la técnica que se utilizará para optimizar los parámetros de nuestra red neuronal.

Las Estrategias Evolutivas son un tipo de algoritmos evolutivos desarrollados por Ingo Rechenberg y Hans-Peter Shwefel en los años 60 (Rechenberg, 1973). La primera estrategia evolutiva (1+1)-EE consistía en un padre y un hijo los cuales competían para saber cuál es el mejor individuo y este se mantenía en la siguiente generación. Después Rechenberg modificó esta estrategia para tener más de un padre, pero seguía generando sólo un hijo, donde cualquier padre puede ser reemplazado en caso de no ser apto, creando así una población de individuos en el espacio de búsqueda. Años después Shwefel desarrolló dos estrategias evolutivas (Schwefel, 1981), la primera es con el reemplazo por exclusión $(\mu + \lambda)$-EE donde los hijos y los padres compiten por ser incluidos en la nueva generación. Y la segunda es con el reemplazo por inserción $(\mu, \lambda)$-EE donde solo los hijos compiten

para ser incluidos en la siguiente generación, los padres menos aptos serán eliminados.

La Programación Evolutiva tiene sus orígenes en los años 60s con el investigador Lawrence Fogel (Fogel, 1966), en este tipo de técnica los individuos se representan con vectores que están compuestos por un estado actual, un símbolo de entrada del alfabeto utilizado, el valor del nuevo estado y un símbolo de salida. Con este vector se genera un autómata de estados finitos. Una característica importante es que la programación evolutiva no incluye el proceso de cruzamiento en su algoritmo, este enfoque se basa en que no puede existir un cruce entre diferentes tipos de especie, a diferencia de un algoritmo genético. La manera en que es creada una nueva generación es solo utilizando el proceso mutación.

La Programación Genética es una técnica que comparte la mayoría de las características de los algoritmos genéticos, tienen una población inicial, operadores de selección, de mutación y de cruzamiento. Pero la gran diferencia radica en que los algoritmos genéticos representan la solución en cadenas codificadas, mientras que en la programación genética se utilizan programas de computadora o esquemas de lenguaje de computadora (representados con árboles) como soluciones de los problemas. La programación genética evalúa a los individuos, en este caso programas, con una función de ajuste la cual evalúa la capacidad que tiene uno de los individuos en resolver el problema en cuestión (Koza, 1992).

Evolución Diferencial. Esta técnica es desarrollada por Storm y Price (Storn, 1997). Los individuos se codifican en forma de vectores de forma aleatoria que evolucionaran para explorar un mayor espacio de búsqueda. Usa los procesos de selección, mutación y cruzamiento. Su funcionamiento se centra en modificar le valor de mutación que se usa para generar los nuevos individuos una vez hecho esto se realizan los procesos de cruzamiento.

Como ya se mencionó los algoritmos genéticos son un tipo de algoritmos evolutivos, que sirven como una técnica de búsqueda estocástica que ha sido utilizado con resultados satisfactorios.

Los algoritmos genéticos trabajan sobre cadenas, en estas cadenas se codifican los valores objetivos, a este tipo de codificación se le llama genotipo, este genotipo debe de tener un tamaño finito en un alfabeto finito. Las formas más comunes de representación son cadenas binarias, de valores reales o de codificación gris.

Uno de los principales investigadores este tipo de técnica fue John Holland (Holland, 1992), quien propuso un Algoritmo Genético simple el con la siguiente estructura:

```
Inicializar población;
Evaluar población;
Mientras que el criterio de terminación no sea alcanzado;
 Seleccionar soluciones para la siguiente población;
 Ejecución del cruzamiento;
 Ejecución de la mutación;
 Evaluar población;
```

Este fue el primer algoritmo del cual se generan una serie de variaciones, pero en general se pueden identificar tres procesos básicos; selección, cruzamiento y mutación.

Iniciar la población se refiera a crear un conjunto de soluciones iniciales. La cantidad de individuos será constante durante toda la ejecución del algoritmo. Cada individuo es codificado es una cadena binaria, es decir, valores de 0 o 1.

Evaluar la población es aplicar una función objetivo sobre cada uno de los individuos generados. Esta función tiene como fin evaluar cada cadena binaria para evaluar a cada individuo

En el proceso de selección, se pretende encontrar a los mejores individuos que cuya aplicación en el problema nos acerca a una solución óptima. Un algoritmo genético puede usar diferentes procesos de selección entre los cuales se encuentran (Goldberg, 1989):

| | |
|---|---|
| Selección por ruleta | En este tipo de selección la probabilidad de que un individuo sea seleccionado es proporcional a la diferencia entre su aptitud y la de sus compañeros. |
| Selección por torneo | La población se divide en grupos y los miembros de cada grupo compiten entre sí. Al final solo es elegido un individuo de cada grupo. |
| Selección proporcional | En esta selección los individuos más aptos tienen la mayor probabilidad de ser seleccionados. |

En el proceso de cruzamiento se seleccionan dos individuos para que una parte de su genotipo sea intercambiado. A estos individuos se les conoce como *padres*, que producen a un nuevo individuo que suele denominarse *hijo*. Si $l$ la longitud de la cadena codificada, el punto de cruzamiento puede tomar valores en un rango de [1, $l$ -1]. $P_c$ es la probabilidad de que el proceso de cruzamiento sea realizado, con un rango que va de 0 a 1.

En el proceso de mutación consiste en invertir los valores binarios de la cadena de forma aleatoria a los hijos generados durante el proceso de cruzamiento. Esto se hace para generar un espacio de soluciones más amplio. $P_m$ es la probabilidad de que el proceso de mutación sea realizado, con un rango que va de 0 a 1.

Todos estos procesos son repetidos hasta que se cumplan los criterios de paro del algoritmo, y cada iteración que se realiza se le llama ciclo generacional.

Del algoritmo genético simple se han desarrollado variaciones, algunos implementan diferentes tipos de selección, o la forma en que se aplican los procesos de cruzamiento y mutación.

Considerando que los parámetros libres de la red neuronal son valores reales, que pueden codificarse en binario pesado y aplicar los operadores correspondientes en binario, se ha optado por aplicar un algoritmo genético. No es necesario aplicar operadores de reparación, porque todas las cadenas binarias son cadenas válidas. Esto no sería siempre cierto si se usara, por ejemplo, una codificación IEEE754.

# Capítulo 3. Desarrollo

## 3.1 EPWavenets

De manera análoga al funcionamiento de las RNFBR presentadas en el Capítulo 2, las *EPWavenets* utilizan funciones wavelets como una opción a las FBR. Las funciones wavelets tienen propiedades que no tienen las FBR. Por ejemplo, el soporte compacto que es una propiedad relevante toda vez que permite capturar información local de una función, y no interferir con la aproximación global. Las FBR también permiten identificar información local, sin embargo, el número de opciones (de funciones) se restringe a algunas de ellas como son: la función gaussiana, la función cuadrática, la función cuadrática inversa, y los splines. Otra propiedad relevante de las funciones wavelets obtenidas con FORP's es la posibilidad de controlar los momentos de desvanecimiento, definidos como:

$$\int x^k \psi(x) dx = 0, \qquad (24)$$

para $k = 1,2, \dots, p$

y que cuanto mayor sea el valor de $p$ los wavelets son "más suaves", lo que facilita la aproximación de funciones que tienen varias derivadas. Los wavelets que maximizan los momentos de desvanecimiento para un soporte compacto dado corresponden a las funciones propuestas por Daubechies (Daubechies, 1992), y que están incluidos dentro de las parametrizaciones de los FORP (Herrera, 2011).

Nótese que el uso de wavelets en redes neuronales es un tema ya abordado (Pourtaghi, 2012; Zhang, 1992; Ypma, 1997) sin embargo, la principal aportación de las *EPWavenets* incluye el uso de wavelets paramétricos optimizados evolutivamente para adaptarlas a un contexto particular, y a su vez mantener la adaptabilidad para aproximar diferentes funciones.

En otros trabajos con *wavenets*, a diferencia de las *EPWavenets*, se usan wavelets sin soporte compacto o con una función wavelet $\psi(x)$ fija. En (Pourtaghi, 2012) por ejemplo, se usa la función wavelet $\psi(x) = -e^{-\frac{x^2}{2}} cos(5x)$ con parámetros de escalamiento y dilatación fijos, y en (Ypma, 1997) y (Zhang, 1992) se usa la función $\psi(x) = -xe^{-\frac{1}{2}x^2}$ conocida como "Derivada-Gaussiana".

En el caso de las *EPWavenets* en lugar de usar funciones con expresión analítica, se aproximan funciones wavelets con valores discretos, para lo cual se usa el *Algoritmo en Cascada* (Mallat, 1989) y filtros ortogonales paramétricos, lo cual nos da una amplia gama de posibles funciones. A manera de ejemplo se puede mencionar que con el valor de $\alpha = \frac{\pi}{4}$ en la ec. ( 25 ) se obtienen filtros de reconstrucción perfecta (filtros de Haar) con la cual se puede a su vez aproximar la

función wavelet de Haar. En la Figura 11 se muestra la wavelet de Haar en forma discreta obtenida con los filtros de Haar y el *Algoritmo en Cascada* con una aproximación de $16$ valores y en donde el soporte es $[0,1]$.

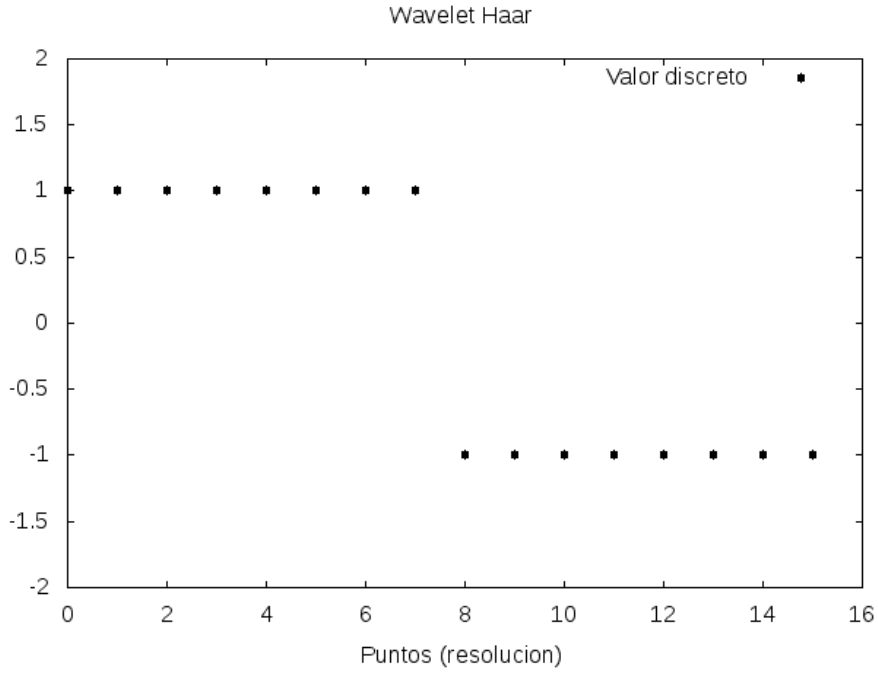



Figura 11. Aproximación del wavelet de Haar con $\alpha = {}^{\pi}/_{4}$ y una aproximación con 16 puntos

La idea fundamental de las redes neuronales *EPWavenets* consiste en incluir a las funciones wavelets generadas a partir de filtros paramétricos (ver ec. ( 21 )y ec.( 22 )) como se hizo en el ejemplo anterior con la wavelet de Haar, y usar la ec. ( 2 ) para aproximar funciones mediante una combinación lineal de las mismas. Adicionalmente se aplican algoritmos evolutivos para determinar el conjunto óptimo de parámetros libres que incluyen a los parámetros de la arquitectura de la red neuronal y de los filtros paramétricos. El uso de wavelets y redes neuronales ya ha sido abordado en otros trabajos. En (Pourtaghi, 2012) por ejemplo, se optimizan los pesos $w_i$ y se dejan fijos los parámetros de traslación y escalamiento, lo cual restringe demasiado la capacidad de aproximar con wavelets. En (Zhang, 1992) se propone usar un método de gradiente para optimizar los parámetros $w_i$ junto con los parámetros $a_i$ y $b_i$ pero la desventaja es que para aplicar métodos de gradiente se requiere conocer la función analítica de $\psi(x)$, misma que además debe ser derivable, algo que restringe la variedad de funciones wavelets. Al usar un reducido número de funciones wavelets, con derivada y expresión analítica conlleva a incluir un mayor número de neuronas cuando la función a aproximar no es suave, en otras palabras se dificulta la aproximación y se complica la arquitectura de la red neuronal.

Es conocido que determinar el conjunto de parámetros óptimos de los filtros paramétricos para aproximar una función es un problema no convexo (Tewfik, 1992). Entonces, dada la complejidad de calcular los parámetros óptimos de los filtros y de los parámetros $a_i$, $b_i$, $w_i$ de la red neuronal, se propone usar algoritmos

evolutivos, por lo que podemos decir que las *EPWavenets* son la combinación de redes neuronales, wavelets y algoritmos evolutivos:

*EPWavenet = Redes Neuronales + Wavelets + Algoritmos Evolutivos*

La Figura 12 ilustra una red neuronal *EPWavenet* con algunos parámetros que involucra, en este caso se muestran los pesos sinápticos $w_i$, los parámetros de traslación $b_i$, y los parámetros de escalamiento $a_i$. La relación entre las funciones wavelets y los filtros de reconstrucción perfecta se listan en las ec. ( 21 ) y ec. ( 22 ).

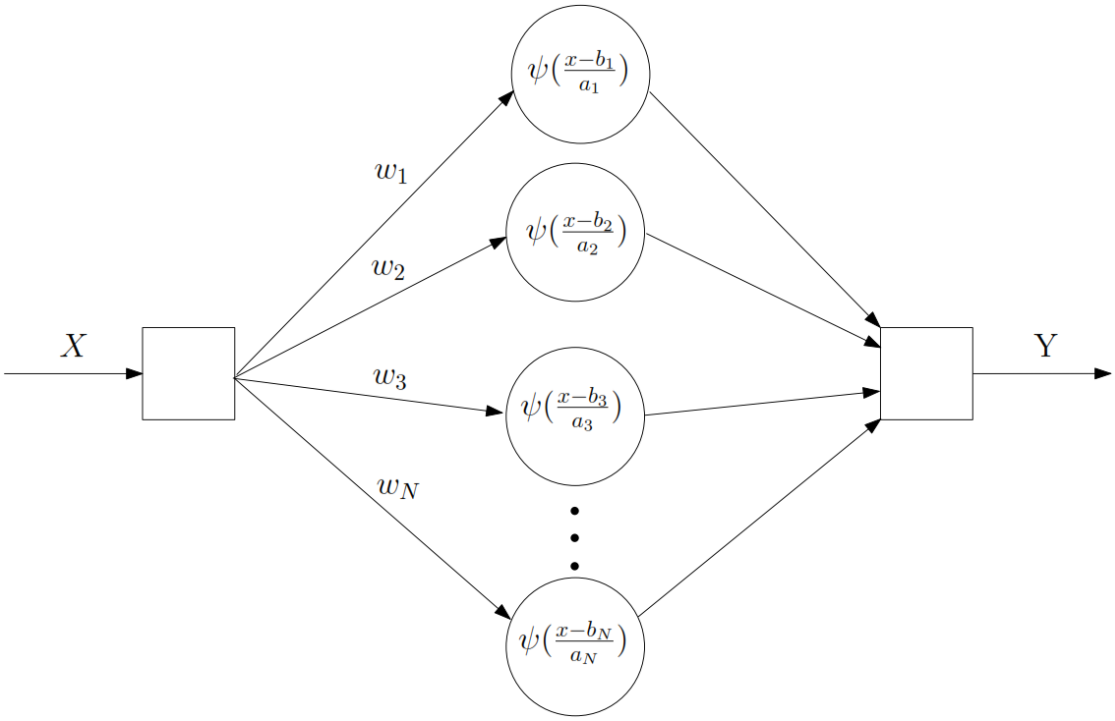



Figura 12. Red neuronal EPWavenet

La función que minimiza el algoritmo evolutivo es el error de aproximación entre la salida $Y$ de la *EPWavenet* y los valores de la función objetivo $f(x)$ obtenidos con la entrada $X$. Tanto $X$ como $Y$ están en la misma dimensión $n$, así que el error de aproximación (cuadrático medio) está dado por la ec. ( 11 ).

La aproximación con *EPWavenets* puede ser suficientemente pequeña de acuerdo con el número de funciones involucradas, y con los valores apropiados de los parámetros de escalamiento y dilatación. Matemáticamente se expresa así:

$$|(\sum_{i=1}^{N} w_i \psi(\frac{x-b_i}{a_i})) - f(x)| < \varepsilon \qquad (25)$$

para $\varepsilon > 0$.

En particular para un filtro de longitud $4$ se tiene un parámetro $\alpha$, y con una arquitectura de $N$ neuronas se generan $3N$ parámetros adicionales que corresponden a los valores de $a_i$, $b_i$, $w_i$, con $i = 1, 2, \dots, N$ que da un total de 3N+1 parámetros libres.

## 3.2 Aproximación con las funciones wavelets generadas con filtros paramétricos

En los experimentos se realiza la aproximación de funciones wavelets a partir de parametrizaciones de filtros de longitud $L = 4$ y $L = 6$. Con $A$ muestras de aproximación para las funciones wavelet $\psi(x)$ con un soporte compacto en el intervalo $[0, \frac{L}{2} - 1]$. Para valores de $x$ dentro del soporte compacto, la mejor aproximación posible para $\psi(x_r)$ está dada por la $r$ −ésima muestra discreta, donde $r = \lfloor x_r * A/(N-1) \rfloor$. Obviamente, para valores fuera del soporte $\psi(x) = 0$. Es importante remarcar que con las consideraciones anteriores, el valor aproximado de la función wavelet es devuelto como si se tuviese la función analítica de $\psi(x)$ y para cualquier valor real de x.

## 3.3 Configuración del algoritmo genético

Para efectos de calcular el valor óptimo de los parámetros $\alpha$ y $\beta$ según la longitud de los filtros paramétricos utilizados se usa un algoritmo evolutivo. Existen varios algoritmos evolutivos que pueden usarse con las *EPWavenets*. En los experimentos realizados se optó por usar un algoritmo genético no tradicional (Kuri, 1999) para codificar los parámetros $\alpha$ y $\beta$ en cada uno de los individuos. Este algoritmo genético se caracteriza por:

- Una población con $P$ individuos, en donde $P$ es par, y una poblacion temporal de tamaño $2P$
- Cruzamiento en parejas entre el $i$-ésimo y el $(P-1-i)$-ésimo individuos, en donde $i \in [0, \frac{P}{2} - 1]$. Los individuos se ordenan según la medida de aptitud
- Selección determinista, en donde los $P$ individuos más aptos de la poblacion temporal pasan a la siguiente generación.
- Cruzamiento anular en un solo punto

Se utilizó una codificación en binario pesado con $q = 29$ bits de precisión por cada parámetro $\alpha$ y $\beta$.

Si $B_{\alpha\beta}$ es el valor en binario pesado de $\alpha$ o $\beta$, entonces para generar valores $V_{\alpha\beta}$ para $\alpha$ o $\beta$ en un rango de $[0, 2\pi)$ se usó la ec.( 26 ):

$$V_{\alpha\beta} = 2\pi \frac{B_{\alpha\beta}}{2^q - 1} \qquad (26)$$

Para codificar los valores de $w$, $a$ y $b$ se usaron $q = 28$ bits de precisión, y un bit de signo. A partir de su valor en binario pesado con signo $B_{wab}$ se obtuvieron valores $V_{wab}$ en el intervalo $(-8192, 8192)$ mediante la ec. ( 25 ):

$$V_{wab} = 8192 \frac{B_{wab}}{2^q - 1} \qquad (27)$$

Los valores de las constantes de las ec. ( 26 ) y ec. ( 27 ) se determinaron experimentalmente, como puede revisarse en (Ramírez, 2014).

## 3.4 Funciones de prueba

En los experimentos se consideraron $13$ funciones $f(x)$ cuyos valores se normalizaron (valores en el intervalo $[0,1]$) usando la fórmula:

$$Vn = \frac{Valor\ real - Valor\ mínimo}{Valor\ máximo - Valor\ mínimo} \qquad (28)$$

La Tabla 1 resume la información de cada función $f(x)$, en donde en la primera columna se muestra el nombre de la función, en la segunda se muestra un idenficador $f\#$, y en la tercera columna se indica el número de muestras o puntos para la función $f(x)$ discretizada (para conocer más las funciones objetivo revisar el Anexo A. Funciones objetivo). El origen de nuestras funciones es el repositorio del software VRA (Kononov, 2010).

Tabla 1. Funciones $f(x)$ del repositorio y sus características

| **Nombre** | **Brownian Motion** | **Dow Jones** | **ECG** | **Henon** | **Ikeda** | **Laser** | **Logistic** |
|---|---|---|---|---|---|---|---|
| **f#** | f1 | f2 | f3 | f4 | f5 | f6 | f7 |
| **Número de muestras (M)** | 1000 | 1000 | 9000 | 5000 | 5000 | 5000 | 1000 |

| **Nombre** | **Lorenz** | **Rossler** | **Seno** | **Seno con Ruido** | **Sun Spots** | **White Noise** |
|---|---|---|---|---|---|---|
| **f#** | f8 | f9 | f10 | f11 | f12 | f13 |
| **Número de muestras (M)** | 3500 | 5000 | 1000 | 1000 | 280 | 1000 |

## 3.5 Esquema de aproximación

En esta sección se presenta el esquema de aproximación de funciones con EPWavenets, el esquema se organizó de acuerdo a la Figura 13. Para fines explicativos, el esquema se ha dividido en etapas enumerados del 1 al 14.

A continuación se describen cada una de esas etapas.

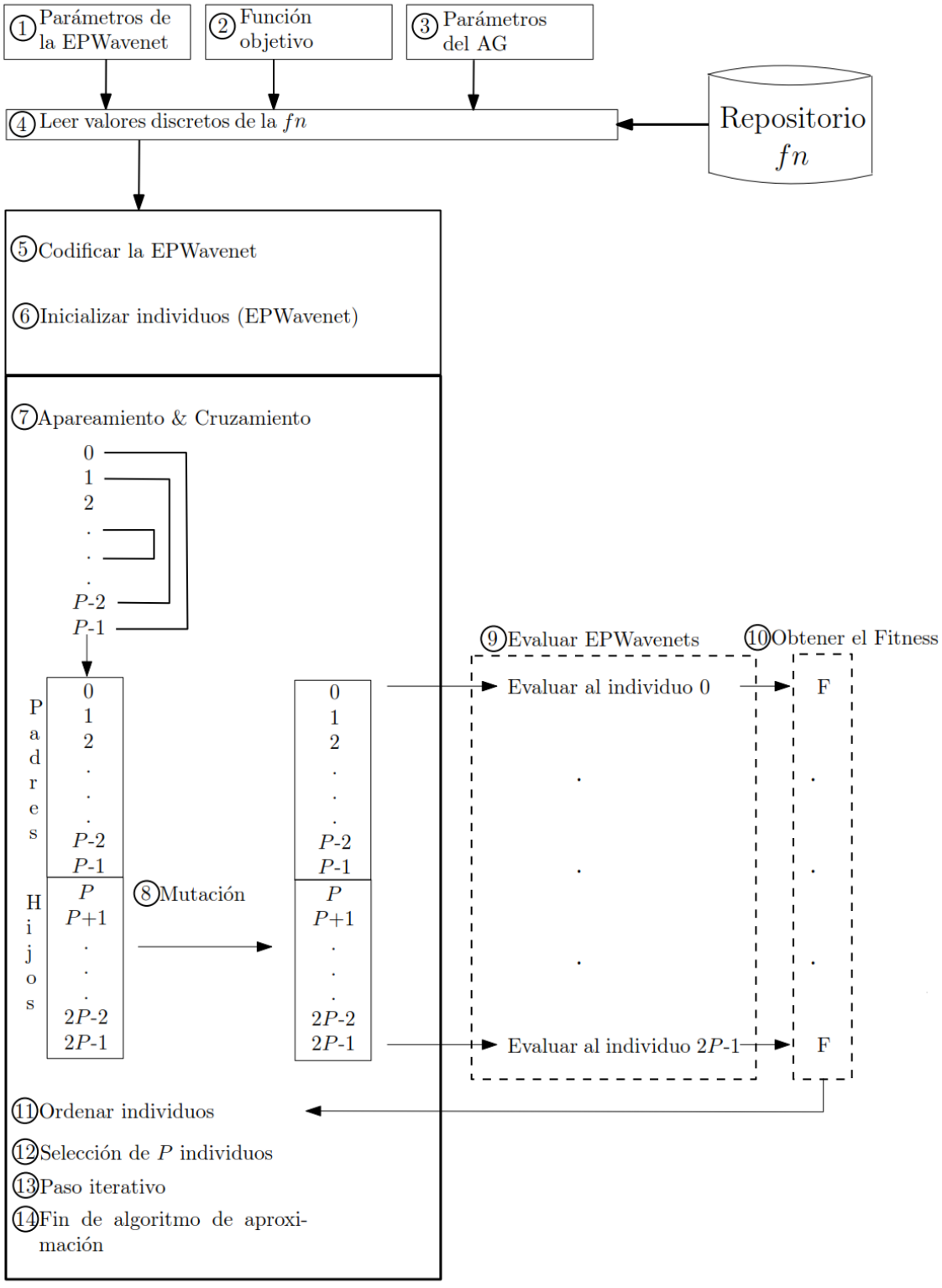



Figura 13 Esquema de aproximación de la EPWavenet

**Etapa 1.** Parámetros de la EPWavenet

El usuario debe de configurar los parámetros de la EPWavenet que son:

- Número de neuronas del modelo
- Longitud del filtro a utilizar

**Etapa 2.** Función Objetivo

Indicar la función objetivo a aproximar.

**Etapa 3.** Parámetros del Algoritmo Evolutivo

Configurar los parámetros del Algoritmo Genético:

- Probabilidad de mutación
- Probabilidad de cruzamiento
- Numero de generaciones
- Tamaño de la población

**Etapa 4.** Leer valores discretos de la función

El modelo obtiene la función objetivo del repositorio de funciones discretas.

**Etapa 5.** Codificar la EPWavenet

Se codifican los valores de cada EPWavenet

$\alpha|a_1 a_2 \dots a_N|b_1 b_2 \dots b_N|w_1 w_2 \dots w_N$ para un filtro de longitud 4

$\alpha\beta|a_1 a_2 \dots a_N|b_1 b_2 \dots b_N|w_1 w_2 \dots w_N$ para un filtro de longitud 6

**Etapa 6.** Inicializar individuos

Se inicializan los individuos, es decir, se inicializan los valores de cada EPWavenet.

**Etapa 7.** Apareamiento y Cruzamiento

Se realiza un apareamiento del individuo i con el individuo P-1-i, este proceso se realiza mediante un cruzamiento anular con probabilidad $P_c$ .

Al final de este paso se tendrá una población de 2P individuos donde la primera mitad de elementos son los individuos padres y la segunda mitad de elementos son los individuos hijos.

**Etapa 8.** Mutación

Se aplica con probabilidad $P_m$ el proceso de mutación a cada bit de cada individuo, y solo a los P individuos hijos en cada generación.

**Etapa 9.** Evaluar EPWavenets

Se genera una EPWavenet con los individuos resultantes de la Etapa anterior, la evaluación de la EPWavenet da como resultado una función de aptitud F, para nuestro caso esta función es el RMS.

**Etapa 10.** Obtener función de aptitud

Se obtiene todas las funciones de aptitud generadas en la Etapa 9.

**Etapa 11.** Ordenar individuos

Los individuos se ordenan del más apto al menos apto, según los valores de la función de aptitud obtenida.

**Etapa 12.** Selección de individuos

Dados 2P individuos identificados como $P_i$ con $i = 0,1,2 \dots ,2P - 1$, y una vez ordenados del más apto $P_0$ al menos apto 2P-1 se seleccionan los P más aptos que pasarán a la siguiente generación.

**Etapa 13.** Etapa (paso) iterativa

Mientras no se excedan el número máximo de generaciones las generaciones regresar al paso 7.

**Etapa 14.** Fin del esquema de aplicación

# Capítulo 4. Resultados experimentales

Se realizaron varios experimentos con la intención de:

- Explorar la capacidad de aproximar funciones de energía finita usando funciones wavelets generadas con filtros ortogonales de reconstrucción perfecta, en una arquitectura de red neuronal denominada *EPWavenet*.
- Comparar el desempeño de funciones wavelets en *EPWavenets* y funciones gaussianas en una arquitectura de RNFBR al aproximar y predecir funciones discretas.

Los experimentos permiten afirmar que es posible aproximar funciones con *EPWavenets*, inspirados en la arquitectura de redes neuronales de funciones de base radial con una sola capa de neuronas, y el uso de funciones wavelets que no requieren conocer su expresión analítica como funciones base en el modelo de red neuronal. Además se puede afirmar que el desempeño de *EPWavenets* es superior al de otras arquitecturas que incluyen funciones de base radial.

Las *EPWavenets* usaron un algoritmo genético configurado de la siguiente manera: el número de generaciones fue $G = 2000$, una población de $P = 50$ individuos, una probabilidad de cruzamiento $P_c = 0.97$ y una probabilidad de mutación $P_m = 0.02$. El tiempo promedio de ejecución del algoritmo genético con los parámetros anteriores para un conjunto de datos de entrada de $M = 1000$ es de $11$ minutos, para $M = 5000$ es de $13$ minutos y para $M = 9000$ es de $16$ minutos, corriendo sobre servidores Sun Fire $X2270$ con procesadores Xeon de $16$ núcleos y $32$ GB de RAM con GNU/Linux Debian.

La codificación de una *EPWavenet* con N neuronas en un individuo del algoritmo genético se realizó de la siguiente manera:

- Para el caso de un filtro de longitud $4$ se tienen un solo parámetro $\alpha$, y la codificación en un individuo es así:

$$\alpha|a_1 a_2 \dots a_N|b_1 b_2 \dots b_N|w_1 w_2 \dots w_N$$

  El individuo completo se divide de la siguiente manera:
  - La primera sección de bits para el valor del parámetro $\alpha$ obtenido de la ec. ( 26 ),
  - las siguientes dos secciones de bits para los valores de los parámetros de $a_1 a_2 \dots a_N$ y $b_1 b_2 \dots b_N$, respectivamente, obtenidos de la ec. ( 27 ),
  - la última sección de bits para los valores de los parámetros $w_1 w_2 \dots w_N$ obtenidos de la ec. ( 27 ).

- Para el caso de un filtro de longitud $6$ se tienen dos parámetros $\alpha$ y $\beta$, y la codificación en un individuo es así:

$$\alpha\beta|a_1a_2\ldots a_N|b_1b_2\ldots b_N|w_1w_2\ldots w_N$$

El individuo completo se divide de la siguiente manera:

- La primera sección de bits para los valores de los parámetro $\alpha$ y $\beta$ obtenidos de la ec. ( 26 ),
- las siguientes dos secciones de bits para los valores de los parámetros de $a_1a_2\ldots a_N$ y $b_1b_2\ldots b_N$, respectivamente, obtenidos de la ec. ( 27 ),
- la última sección de bits para los valores de los parámetros $w_1w_2\ldots w_N$ obtenidos de la ec. ( 27 ).

## 4.1 Experimento 1

En el primer experimento se busca determinar el mejor modelo de la *EPWavenet* para una función específica $f\#$ , con datos de entrada en una dimensión ($n = 1$), variando el número de neuronas $N = 1,2,\ldots,10$, y el tipo de filtro (wavelet) con $L = 4$ para un filtro de longitud 4, y $L = 6$ para un filtro de longitud 6, ver ec. ( 21 ) y ( 22 ). El criterio para elegir la mejor arquitectura fue minimizar el error cuadrático medio (ver ec. ( 11 )), sobre las $M$ muestras disponibles para cada función discreta. Los resultados se muestran en la Tabla 2 y en la Tabla 3, en donde en la primera columna se indica el valor de $L$ y $N$, y en las demás columnas se indica el tipo de función aproximada identificada por $f\#$. La mejor arquitectura por columna, comparando los resultados de ambas tablas, se ha remarcado en negritas para cada caso de $f\#$.

Tabla 2. Arquitecturas EPWavenets para la aproximación de las funciones $f1$ a $f7$

| **Función** | $f1$ | $f2$ | $f3$ | $f4$ | $f5$ | $f6$ | $f7$ |
|---|---|---|---|---|---|---|---|
| **L , N** | | | | | | | |
| **4,1** | 1.84E-6 | 0.00434 | 0.09305 | 0.19952 | 0.17513 | **2.91E-9** | 1.21E-5 |
| **4,2** | 0.00345 | 0.00903 | 0.07690 | 0.21114 | 0.04699 | 0.03520 | 0.00183 |
| **4,3** | 0.00703 | 0.00869 | 0.02102 | 0.00541 | 0.06779 | 0.01058 | 3.12E-4 |
| **4,4** | 4.06E-4 | 0.00594 | 0.02525 | 0.02009 | 0.01489 | 0.03456 | 1.65E-4 |
| **4,5** | 2.87E-4 | 5.30E-4 | 0.02084 | 0.06891 | 0.00753 | 0.00669 | 2.39E-4 |
| **4,6** | 1.27E-5 | 8.12E-4 | 0.02941 | 0.04353 | 0.01084 | 0.00444 | 4.26E-5 |
| **4,7** | 2.28E-5 | 0.00152 | 0.03750 | 0.07567 | 0.01000 | 0.01069 | **2.20E-10** |
| **4,8** | **1.68E-8** | 0.00649 | 0.01166 | 8.23E-5 | **1.63E-4** | 0.00436 | 2.97E-5 |
| **4,9** | 1.05E-5 | 0.00315 | 0.01743 | 0.05384 | 0.05532 | 0.00418 | 6.67E-5 |
| **4,10** | 3.50E-5 | 0.00133 | 0.03197 | 0.08349 | 0.01160 | 0.01280 | 4.18E-4 |
| **6,1** | 0.00519 | 0.00587 | 0.08905 | 0.12375 | 0.19189 | 0.02997 | 9.47E-4 |
| **6,2** | 0.02138 | 0.00246 | 0.02127 | 0.11266 | 0.00378 | 0.02125 | 2.22E-4 |
| **6,3** | 0.00258 | 0.01542 | 0.03566 | 0.07052 | 0.00833 | 0.02580 | 2.16E-5 |
| **6,4** | 2.69E-5 | 0.00432 | 0.03510 | 0.08667 | 0.01482 | 0.00532 | 1.09E-4 |
| **6,5** | 5.94E-5 | 6.13E-4 | 0.01794 | 0.02906 | 0.04455 | 7.26E-4 | 4.59E-5 |
| **6,6** | 2.91E-4 | **1.53E-4** | 0.02701 | **3.17E-5** | 0.00511 | 0.00200 | 2.47E-4 |
| **6,7** | 0.01714 | 0.01390 | **1.57E-4** | 0.03620 | 3.12E-4 | 0.00145 | 0.00552 |
| **6,8** | 4.34E-5 | 0.00172 | 0.06659 | 0.00166 | 0.01947 | 0.00430 | 0.00331 |
| **6,9** | 5.77E-5 | 0.03970 | 0.01903 | 2.92E-4 | 0.04756 | 0.03121 | 2.07E-6 |
| **6,10** | 2.06E-4 | 0.00434 | 0.06997 | 0.00445 | 0.05882 | 0.00138 | 0.00303 |

Tabla 3. Arquitecturas EPWavenets para la aproximación de las funciones $f8$ a $f13$

| **Función** | $f8$ | $f9$ | $f10$ | $f11$ | $f12$ | $f13$ |
|---|---|---|---|---|---|---|
| **L , N** | | | | | | |
| **4,1** | 0.24306 | **1.18E-5** | 4.62E-6 | **1.28E-16** | 7.23E-4 | 0.00398 |
| **4,2** | 0.07842 | 0.02476 | 0.00285 | 0.00229 | 3.98E-5 | 1.99E-4 |
| **4,3** | 0.01952 | 0.09505 | 0.00107 | 1.22E-4 | 9.59E-7 | 1.47E-7 |
| **4,4** | 0.01685 | 0.01646 | 0.00204 | 5.65E-4 | 1.56E-5 | 1.17E-4 |
| **4,5** | 0.01551 | 0.08655 | 3.58E-5 | 4.49E-4 | 9.48E-5 | 3.63E-4 |
| **4,6** | 0.01015 | 0.00648 | 2.47E-4 | 1.53E-4 | 7.31E-7 | 5.97E-6 |
| **4,7** | 0.01065 | 0.01655 | 1.13E-5 | 3.10E-9 | 1.16E-5 | 1.17E-4 |
| **4,8** | **8.87E-6** | 0.02839 | **1.79E-8** | 3.64E-4 | 1.27E-5 | **5.31E-8** |
| **4,9** | 0.04234 | 4.72E-5 | 2.25E-5 | 8.31E-5 | 9.05E-7 | 2.36E-5 |
| **4,10** | 0.00818 | 0.01212 | 6.54E-4 | 1.08E-5 | 9.22E-6 | 3.08E-5 |
| **6,1** | 0.10162 | 0.06970 | 9.76E-4 | 2.11E-5 | 3.21E-4 | 0.01368 |
| **6,2** | 0.00426 | 0.03568 | 0.01482 | 0.00128 | 1.31E-4 | 0.00266 |
| **6,3** | 0.03497 | 0.05618 | 5.82E-5 | 0.0074 | 0.00109 | 2.02E-4 |
| **6,4** | 0.01528 | 0.04018 | 5.06E-5 | 1.99E-4 | 4.19E-4 | 1.78E-4 |
| **6,5** | 6.33E-4 | 4.11E-4 | 2.84E-5 | 1.39E-4 | 2.10E-4 | 5.51E-4 |
| **6,6** | 0.01911 | 0.00886 | 0.00622 | 3.80E-5 | 0.00100 | 2.23E-4 |
| **6,7** | 0.03061 | 7.14E-4 | 0.01605 | 0.00945 | 2.96E-4 | 1.62E-5 |
| **6,8** | 7.84E-4 | 0.00771 | 0.00101 | 1.34E-4 | **4.42E-7** | 3.69E-4 |
| **6,9** | 3.68E-5 | 0.00212 | 3.27E-4 | 1.20E-4 | 1.90E-6 | 6.36E-6 |
| **6,10** | 3.41E-4 | 0.00229 | 0.00157 | 1.95E-4 | 5.31E-6 | 9.08E-4 |

El número promedio de neuronas requeridas en las *EPWavenets* es $\overline{N} = 5.85$ con una desviación estándar de $\sigma_N = 2.85$. El error promedio de aproximación para los 13 casos fue $4.05E-5$ con una desviación estándar de $\sigma = 6.74E-5$.

En el $69\%$ de los casos la mejor aproximación se obtuvo con filtros de longitud $L = 4$ y en el restante $31\%$ se obtuvo la mejor aproximación con filtros de longitud $L = 6$. Esto se atribuye a la dificultad inherente de optimizar un parámetro $\beta$ adicional.

Como comentario adicional se hace saber que la parametrización con $L = 4$ es un subcaso de la parametrización de $L = 6$, es decir, el mismo resultado que $L = 4$ podría obtenerse con un filtro de $L = 6$ para ciertos valores de $\alpha$ y $\beta$, pero la convergencia del algoritmo genético se hace más lenta. Esta es una de las razones para probar con parametrizaciones de longitud $L = 4$.

## 4.2 Experimento 2

Se comparó el resultado de la aproximación entre la mejor arquitectura *EPWavenet* del Experimento 1 y la aproximación obtenida con una RNFBR optimizada evolutivamente usando una función gaussiana (RNFBRG). Esto

significa haber usado el mismo número de neuronas para cada $f\#$ del Experimento 1 pero diferente función base y diferentes pesos sinápticos.

- Para el caso de la RNFBRG la codificación en un individuo es así:

$$\mu_1\mu_2\ldots\mu_N|\sigma_1\sigma_2\ldots\sigma_N|w_1w_2\ldots w_N$$

El individuo completo se divide de la siguiente manera:

- La primera sección de bits para los valores de los parámetros $\mu_{\mathrm{i}}$ , acorde con la ec. ( 20 ), con *i* = 1,2…,N.
- La segunda sección de bits para los valores de los parámetros $\sigma_{\mathrm{i}}$ , acorde con la ec. ( 20 ), con *i* = 1,2…,N.
- la última sección de bits para los valores de los parámetros $w_1w_2\ldots w_N$ obtenidos de la ec. ( 27 ), en donde N es el número de neuronas.

Los resultados se muestran en la Tabla 4 en donde en la primera columna se muestra el tipo de red neuronal (*EPWavenet* o RNFBR) y en las columnas restantes se muestra el error de aproximación de cada una de las funciones $f1$ a $f13$. La Tabla 4 se ha dividido en 2 renglones por cuestiones de espacio.

Tabla 4. Comparación de EPWavenets y RNFBR con función gaussiana

| **Función** | $f1$ | $f2$ | $f3$ | $f4$ | $f5$ | $f6$ | $f7$ |
|---|---|---|---|---|---|---|---|
| **Red neuronal** | | | | | | | |
| **EPWavenet** | 1.68E-8 | 1.53E-4 | 1.57E-5 | 3.17E-5 | 1.63E-4 | 2.91E-9 | 2.20E-10 |
| **RNFBRG** | 2.86E-7 | 7.86E-4 | 5.92E-4 | 1.96E-4 | 0.04887 | 9.01E-5 | 0.01233 |

| **Función** | $f8$ | $f9$ | $f10$ | $f11$ | $f12$ | $f13$ |
|---|---|---|---|---|---|---|
| **Red neuronal** | | | | | | |
| **EPWavenet** | 8.87E-6 | 1.18E-5 | 1.79E-8 | 1.28E-16 | 4.42E-7 | 5.31E-8 |
| **RNFBRG** | 2.17E-4 | 1.50E-4 | 1.85E-4 | 0.06237 | 8.45E-4 | 1.80E-6 |

En la Tabla 4 se puede observar que en todos los casos la *EPWavenet* mejora la aproximación de la RNFBRG. El error promedio de la aproximación con *EPWavenets* es $4.04E-005$ con una desviación estándar de $6.74E-5$, en tanto que el error promedio de aproximación de la RNFBRG es $9.74E-3$ con una desviación estándar de $2.08E-2$, es decir, el desempeño de las *EPWavenet* es consistentemente mejor que el de las RNFBRG.

## 4.3 Experimento 3

El propósito del Experimento 3 es medir el desempeño de las *EPWavenets* como predictores.

Se generalizó la ec. ( 11 ) al introducir un desplazamiento $k >= 0$ entre el $i$-ésimo dato de entrada en el momento $m$, y el $i$-ésimo dato de salida en el momento $m + k$ que matemáticamente se expresa así:

$$Error(k) = \frac{1}{M}\sum_{m=1}^{m=M}\sum_{i=1}^{n}(x_i(m) - y_i(m+k))^2 \quad (29)$$

De esta forma, la ec. ( 29 ) permite realizar la predicción de la función al establecer la dependencia $y(m+k) = f(x(m))$.

Se compararon las predicciones para $k = 1, \ldots, 100$[1] para la mejor arquitectura *EPWavenet* (ver Experimento 1) y las RNFBRG con el mismo número de neuronas de la *EPWavenet* correspondiente, para las funciones $f1$ a $f13$. Adicionalmente, se compararon a las *EPWavenets* con otro tipo de red similar que en lugar de funciones wavelets y gaussianas usa una función sigmoidal. Esto para explorar su comportamiento como predictor con una capa oculta al considerar que, a diferencia de las funciones wavelets y las funciones gaussianas, la función sigmoidal no concentra su energía en un intervalo dado, y para ciertas funciones esto podría complicar la aproximación.

Los resultados se muestran en la Figura 26 mismas que en cada subfigura comparan a las *EPWavenets* con redes neuronales con funciones gaussianas y redes neuronales con funciones sigmoidales.

En las siguientes figuras se puede apreciar que para Figura 14, Figura 15, y Figura 21 (Brownian Motion, Dow Jones y Lorenz) las *EPWavenets* tienen un mejor desempeño que el resto de las redes neuronales. También se observa que al incrementarse el valor de $k$ el error de aproximación se incrementa paulatinamente.

Algo similar ocurre con la función Figura 22 (Rossler) en donde el error se incrementa conforme el valor de $k$, sin embargo la aproximación con *EPWavenets* se ve superada por el resto de las redes neuronales. Los peores resultados con *EPWavenets* se dan con la función Figura 16 (EGC) que, si bien es cierto, ofrece la mejor aproximación con $k = 0$ y falla con $k > 0$. En tanto que la RNFBRG y la red con función sigmoidal siguen un comportamiento suave e incremental.

Para los casos Figura 17, Figura 18, Figura 19, Figura 20 y Figura 26 (Henon, Ikeda, Laser, Logistic, y White Noise) el comportamiento es prácticamente el mismo para cualquier $k > 1$, es decir, sólo se puede aproximar para $k = 0$, y para $k > 0$ el error de aproximación se dispara de inmediato con cualquier tipo de red neuronal. De hecho, la peor predicción se obtiene con la función sigmoidal que presenta saltos abruptos para diferentes valores de $k$, esto se atribuye a que no puede captar singularidades en las funciones objetivo y toda vez que este tipo de funciones objetivo siguen un comportamiento caótico.

[1] A excepción del caso de $f12$ se usaron el 10% de las muestras, esto es $k = 1, 2, \ldots, 28$.

Para los casos Figura 23, Figura 24 y Figura 25 (Seno, Seno con Ruido y Sun Spots) se observa que todas las redes neuronales se comportan de forma similar y que el valor de predicción fluctúa dependiendo de la periodicidad de las funciones. La peor predicción se logra con funciones sigmoidales, en tanto que la *EPWavenet* tiene un desempeño competitivo respecto a la RNFBRG.

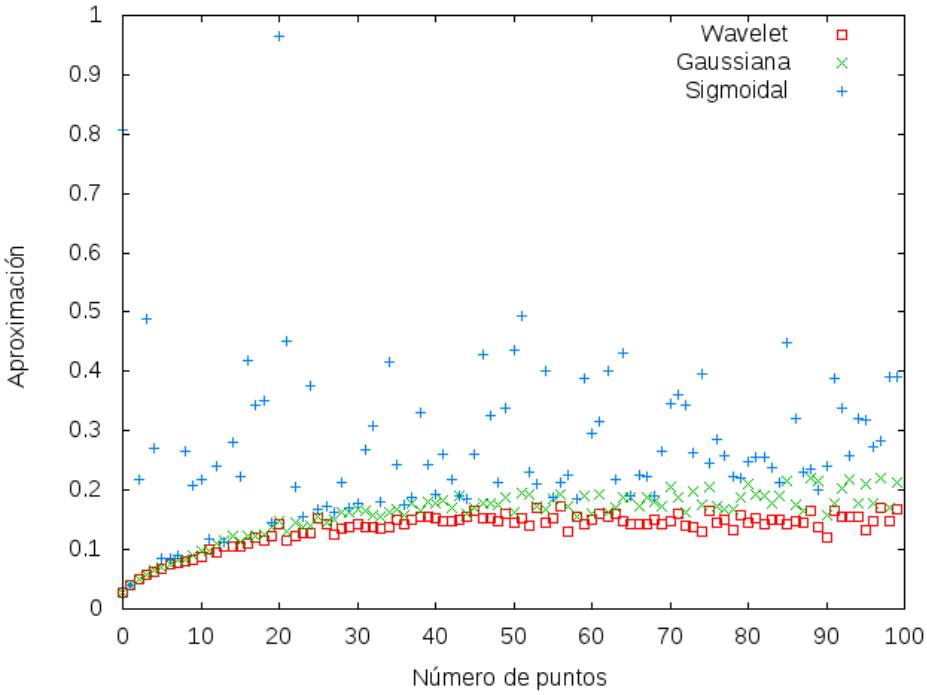



*Figura 14. Predicción de Brownian Motion*

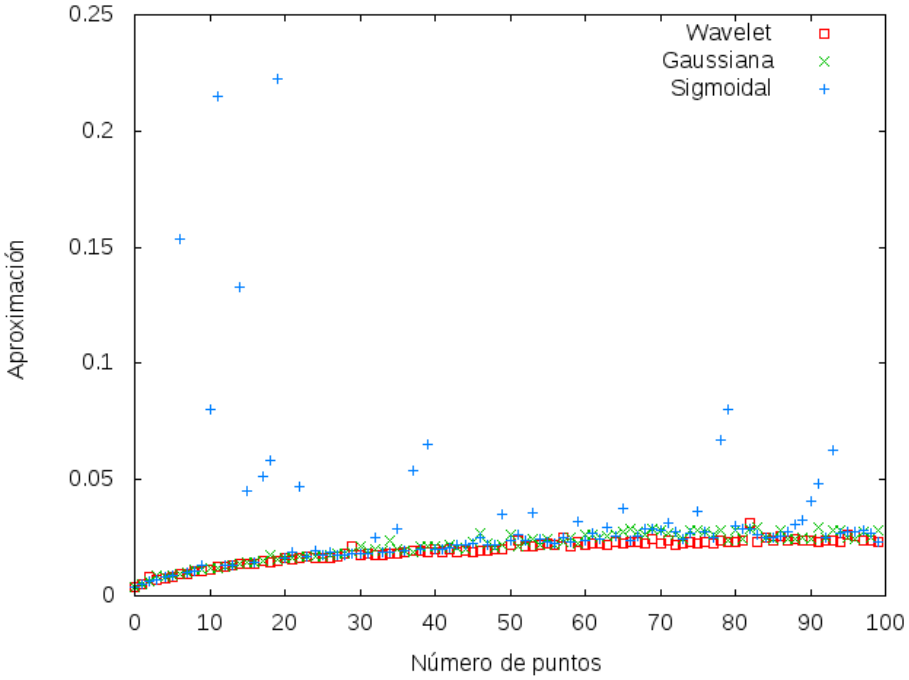



*Figura 15. Predicción de Dow Jones*

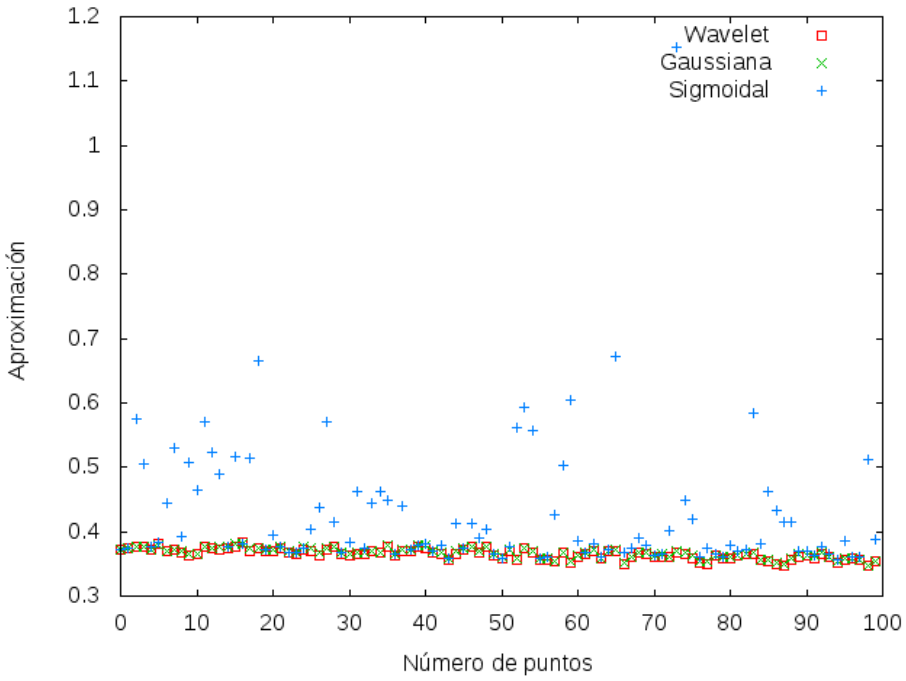



*Figura 16. Predicción de ECG*

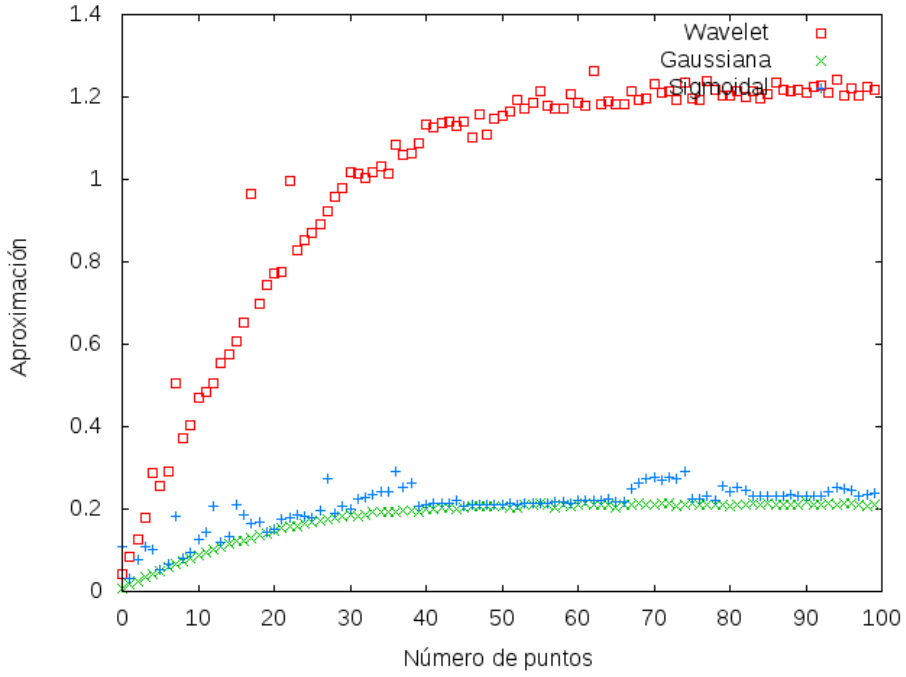



*Figura 17. Predicción de Henon*

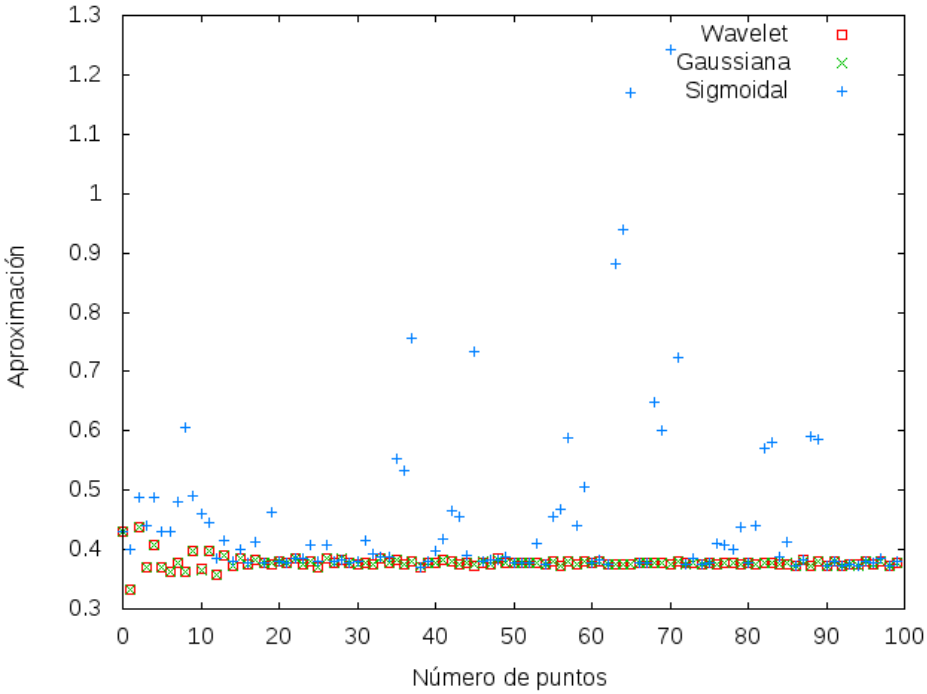



*Figura 18. Predicción de Ikeda*

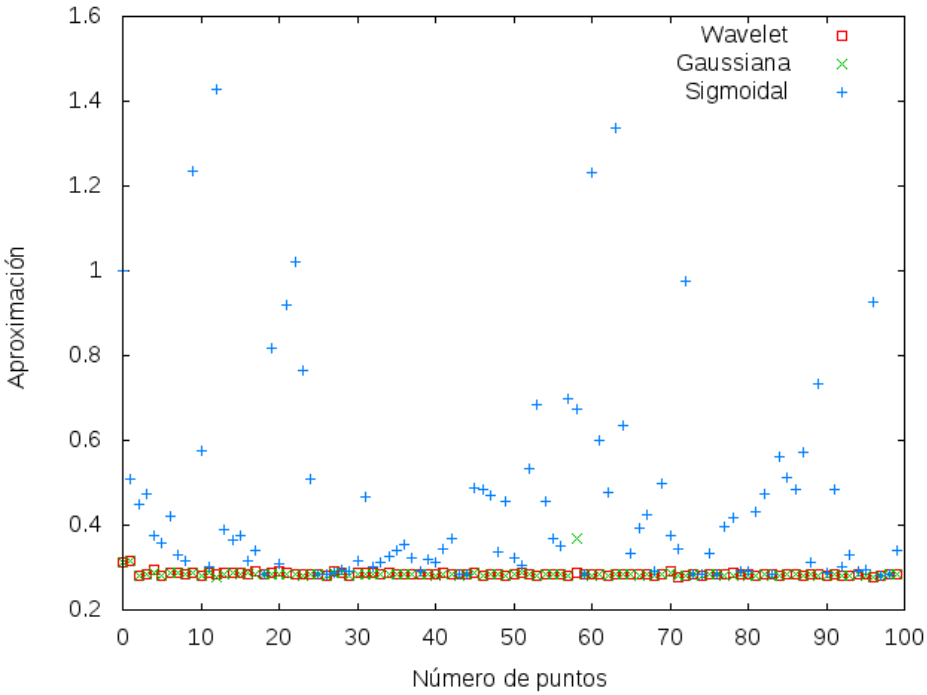



*Figura 19. Predicción de Laser*

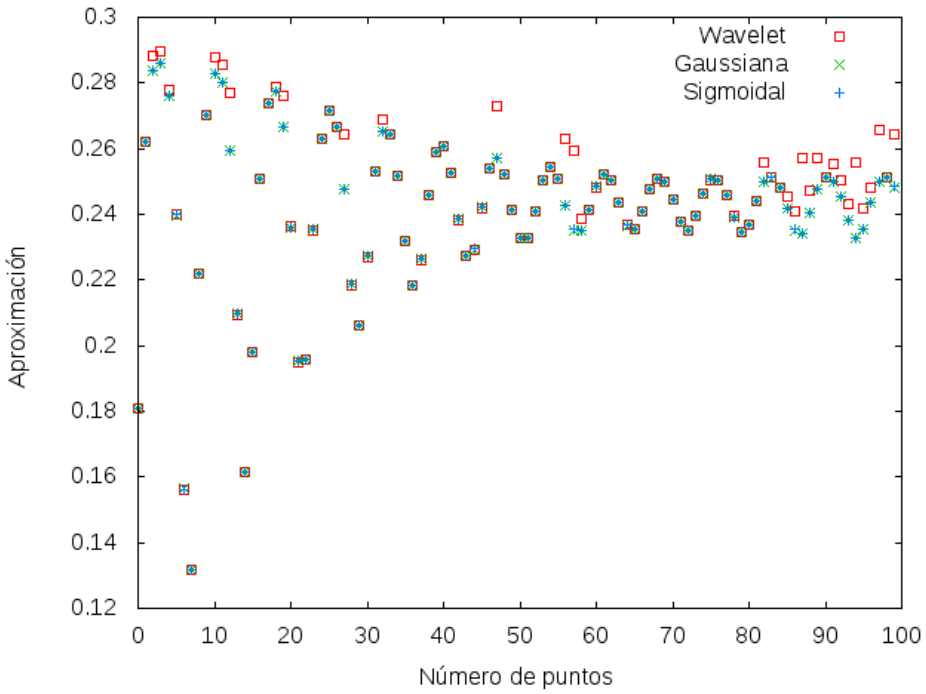



*Figura 20. Predicción de Logistic*

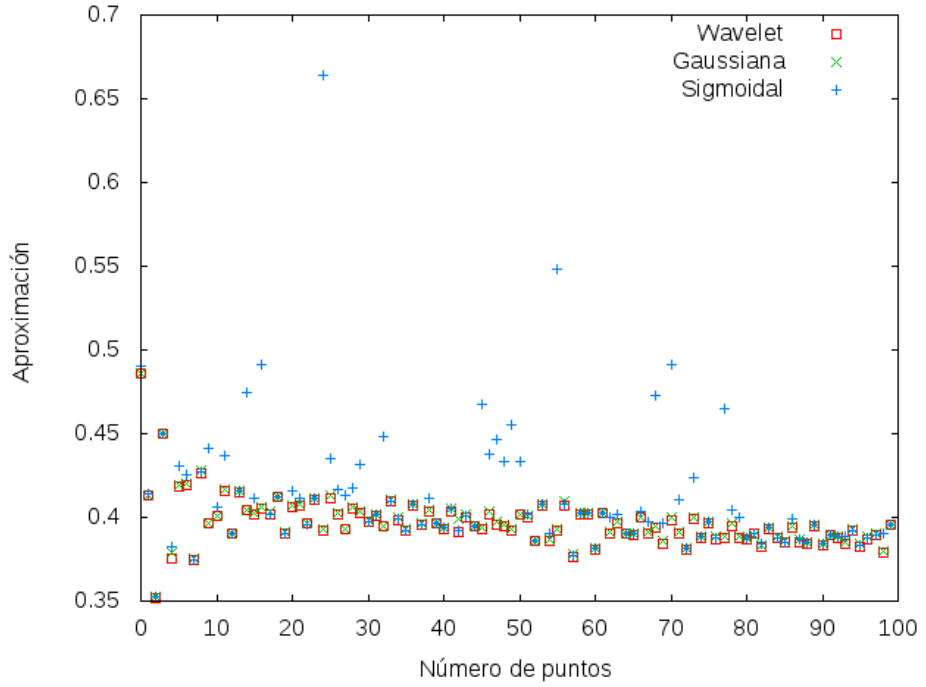



*Figura 21. Predicción de Lorenz*

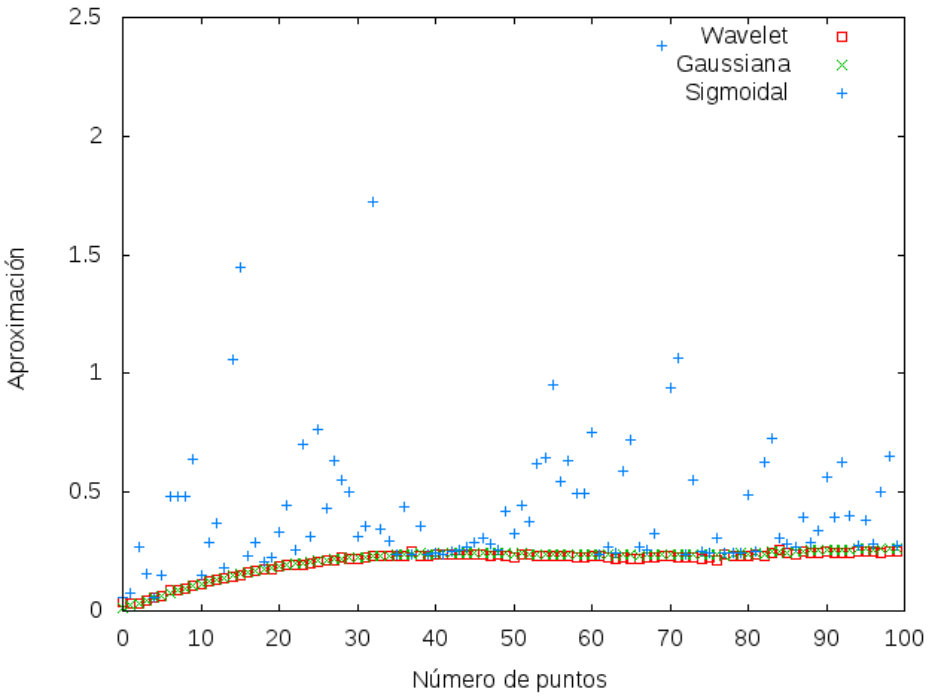



*Figura 22. Predicción de Rossler*

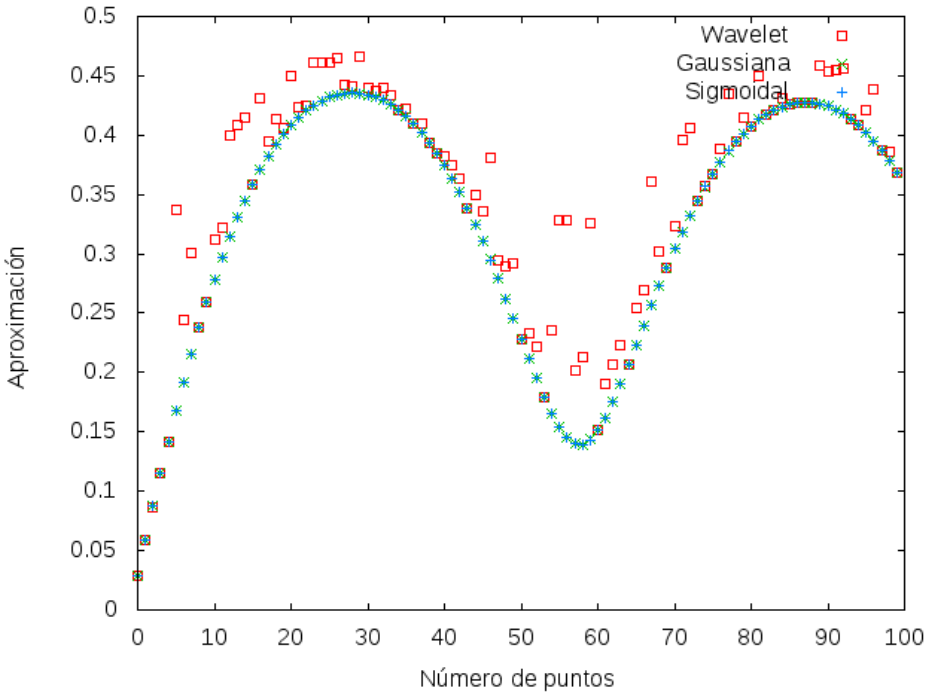



*Figura 23. Predicción de Seno*

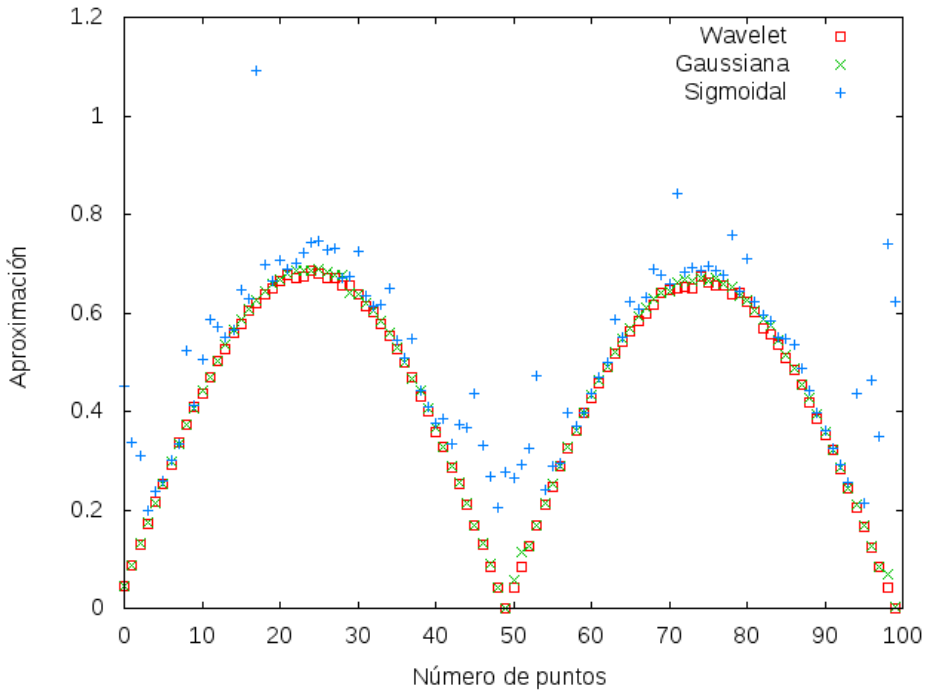



*Figura 24. Predicción de Seno con Ruido*

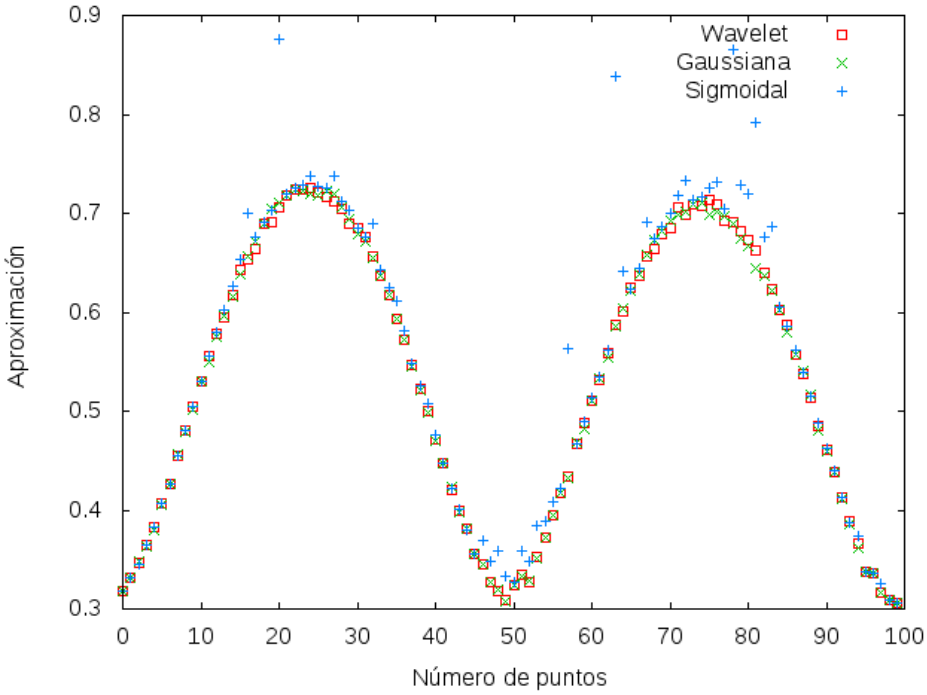



*Figura 25. Predicción de Sun Spots*

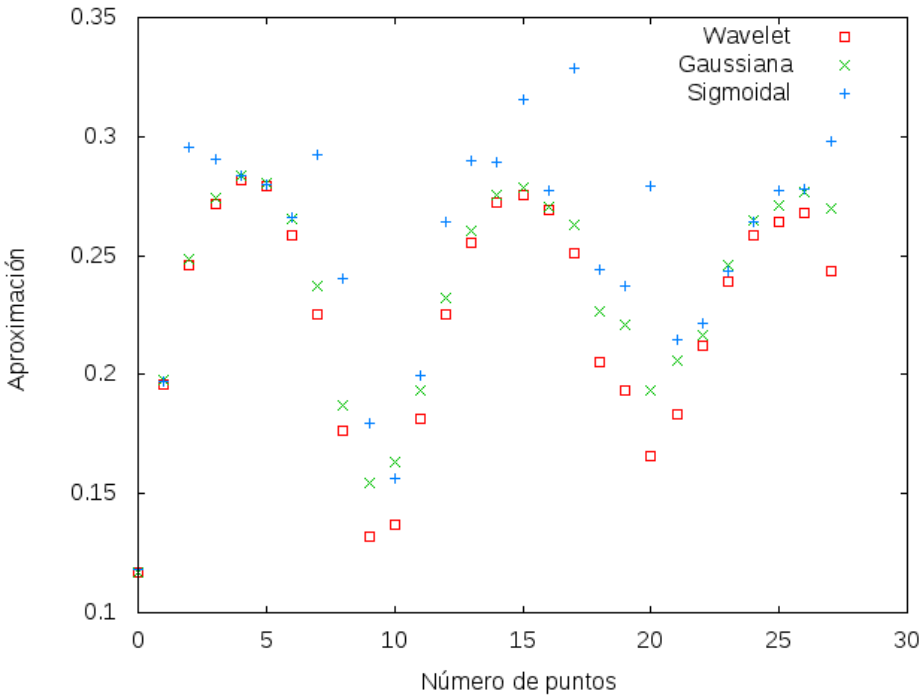



*Figura 26. Predicción de White Noise*

Con los resultados obtenidos con los tres experimentos expuestos en este capítulo podemos concluir que la combinación de redes neuronales y funciones wavelet generadas con filtros ortogonales de reconstrucción perfecta tiene un buen desempeño en comparación con las funciones de base radial, generando mejores resultados en el ámbito de aproximación, mientras que en la predicción se tiene un desempeño comparable con funciones gaussianas y sigmoidales.

# Capítulo 5. Conclusiones y trabajos futuros

En este trabajo de investigación se propuso un modelo de red neuronal denominado *EPWavenet*. Las *EPWavenets* combinan técnicas de cómputo evolutivo, redes neuronales y funciones wavelets obtenidas con filtros paramétricos de reconstrucción perfecta.

Los resultados experimentales sustentan la hipótesis de que es posible usar funciones wavelets para aproximar funciones de energía finita, en una arquitectura de red neuronal.

Fue posible comparar el desempeño de las EPWavenets con otras redes neuronales que usan funciones de base radial sobre un conjunto de funciones de prueba, que en su mayoría representan fenómenos complejos y no lineales.

Se pudo confirmar que es posible usar algoritmos evolutivos en *EPWavenets* para optimizar los parámetros libres a efecto de minimizar el error de aproximación.

Se puede concluir que es posible aproximar funciones discretas con funciones wavelets de las cuales no se tiene su expresión analítica.

Adicionalmente, de los experimentos, se puede concluir que se requiere un número reducido de neuronas con wavelets en tareas de aproximación de funciones, lo cual se atribuye al uso de funciones adaptables mediante su parametrización.

Se concluye que las *EPWavenets* logran un alto grado de adaptabilidad y un desempeño competitivo respecto a otras redes neuronales que involucran funciones de base radial.

Los experimentos permitieron comparar el desempeño de las *EPWavenets* con RNFBRG y se observó que, en cuanto a aproximación se refiere, las *EPWavenets* logran un mejor desempeño.

En cuanto a predicción se refiere, al comparar la predicción de *EPWavenets* con RNFBR con funciones gaussianas y funciones sigmoidales, se concluye que estás últimas tienen un peor desempeño, lo cual se atribuye a la falta de localización temporal de la función sigmoidal, que se extiende por todo el eje real.

De lo anterior se puede asegurar que se han cumplido cada uno de los objetivos planteados en el presente trabajo de investigación.

Se tiene contemplado extender los experimentos para hacer comparaciones con otras funciones de base radial como son la función cuadrática, la función multicuadrática, la función inversa cuadrática, y las funciones *splines*. Además de

usar parametrizaciones de mayor longitud, y plantear el problema de aproximación con *EPWavenets* como un problema multiobjetivo. También se contempla incrementar la dimensión $n$ de los datos de entrada, toda vez que en este trabajo de investigación solo se trabajó con vectores de entrada unidimensionales. Otros trabajos futuros también incluyen aplicar *EPWavenets* en problemas de clasificación y reconocimiento de patrones.

## Agradecimientos

A la Universidad Autónoma Metropolitana, por el uso de los servidores Sun Fire $X2270$ con procesadores Xeon de $16$ núcleos y $32$ GB de RAM con GNU/Linux Debian, obtenidos con el financiamiento del proyecto SEP-PROMEP. Estos servidores están bajo resguardo del doctor Oscar Herrera Alcántara.

## Referencias Bibliográficas

Ambardar, A. (2002). *Procesamiento de señales analógicas y digitales.* Thomson Learning.

Bakshi, B. a. (1992). Wavelets as basis functions for localized learning in a multi-resolution hierarchy. En *International Joint Conference on Neural Networks,IJCNN.* (págs. 140-145).

Briggs, B. C. (2014). *Calculus for Scientists and Engineers.* London: Pearson Education Limited.

Büssow, R. (2007). An algorithm for the continuous Morlet wavelet transform. *Mechanical Systems and Signal Processing*, 2970-2979.

Cammarota, C. (2009). Time series analysis of data from stress ECG. *Communications to SIMAI Congress*.

Chen, S. a. (1991). Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 302-309.

Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. En *Mathematics of Control, Signals, and Systems* (págs. 303–314).

Daubechies, I. (1992). Ten lectures on wavelets. *Society for Industrial and Applied Mathematics.* Philadelphia, PA, USA.

Delyon, B. J. (1995). Accuracy analysis for wavelet approximations. En *IEEE Transactions on Neural Networks 6(2)* (págs. 332–348).

*Deparment of statistics.* (25 de mayo de 2008). Obtenido de University of California: http://www.stat.berkeley.edu/~peres/bmbook.pdf

Dominguez, M. R. (2012). Algoritmos Wavanets con Aplicaciones en la Aproximación de Señales: un Estudio Comparativo. En *Revista Iberoamericana de Automática e Informática Industrial* (págs. 347-358).

Dorf, R. S. (2006). *Introduction to electric circuits.* New York: John Wiley & sons.

Fogel, L. O. (1966). *Artificial Intelligence through Simulated Evolution.* New York: John Wiley & Sons.

Folland, G. (1984). *Real Analysis.* New York: John-Wiley-Interscience.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley.

Halliday, D. J. (2001). *FUNDAMENTOS DE FISICA (VOL. I)*. México: S.L. ALAY EDICIONES.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation.* Prentice Hall.

Hénon, M. (1976). A two-dimensional mapping with a strange attractor. *Commun. Math. Phys*, 69-77.

Hereford, J. R. (2003). Image compression using parameterized wavelets with feedback. *SPIE*, 267-277.

Herrera, O. M. (2011). Aplicación de Algoritmos Genéticos a la Compresión de Imágenes con Evolets. *Sociedad Mexicana de Inteligencia Artificial.*

Holland, J. (1992). *Adaptation in Natural and Artificial Systems.* MIT University Press.

Hsu, H. P. (2000). *Análisis de Fourier.* México: S.A. ALHAMBRA MEXICANA.

Ikeda, K. (1979). Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. *Optics Communications*, 257-261.

Jackson, J. (1999). *Classical Electrodynamics.* USA: Wiley.

Jea-Rong, T. P.-C.-I. (1996). A sigmoidal radial basis function neural network for function approximation. En *IEEE International Conference on Neural Networks* (págs. 496-501).

Kobayashi, K. T. (1994). A wavelet neural network for function approximation and network optimization. *Proceedings of the Artificial Neural Networks in Engineering* .

Kononov, E. (2010). *Institute of Computer Science*. Recuperado el 6 de abril de 2015, de http://www2.informatik.uni-osnabrueck.de/marc/lectures/zra_ss03/prgdat/vra4v2.zip

Koza, J. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge: The MIT Press.

Kuri, A. (1999). *A Comprehensive Approcah to Genetic Algorithms in Optimization and Learning.* Mexico: National Polytechnic Institute.

Kuri., A. A. (2013). *The Best Genetic Algorithm I.* Springer Berlin Heidelberg.

Mallat, S. (1989). Multiresolution approximations and wavelet orthonormal bases of L^2 (R). En *Trans. Amer. Math. Soc. 315(1)* (págs. 69–87).

Mancha, U. d.-L. (2016). *http://www.dsi.uclm.es/ntsa/Series.html.* Recuperado el 20 de marzo de 2016, de http://www.dsi.uclm.es/ntsa/Series.html

Maqsood, I. R. (2004). An ensemble of neural networks for weather forecasting. *Neural Computing & Applications, 13(2)*, 112-122.

Martin, B. A. (2005). *Redes Neuronales y Sistemas Difusos.* Madrid: Alfaomega, RA-MA.

McCulloch, W. P. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 115-133.

Minsky, M. P. (1989). *Perceptrons.* Cambridge: MIT Press.

Ning, J. D. (2008). Wavelet Basis Function Neural Networks for Sequential Learning. *IEEE Transactions on Neural Networks*.

Pourtaghi, A. (2012). Wavelet Neural Network and Wavenet Performance Evaluation in Hydrodynamic Force Prediction due to Waves on Vertical Cylinders. *International Journal of Information and Computer Science*, 187-213.

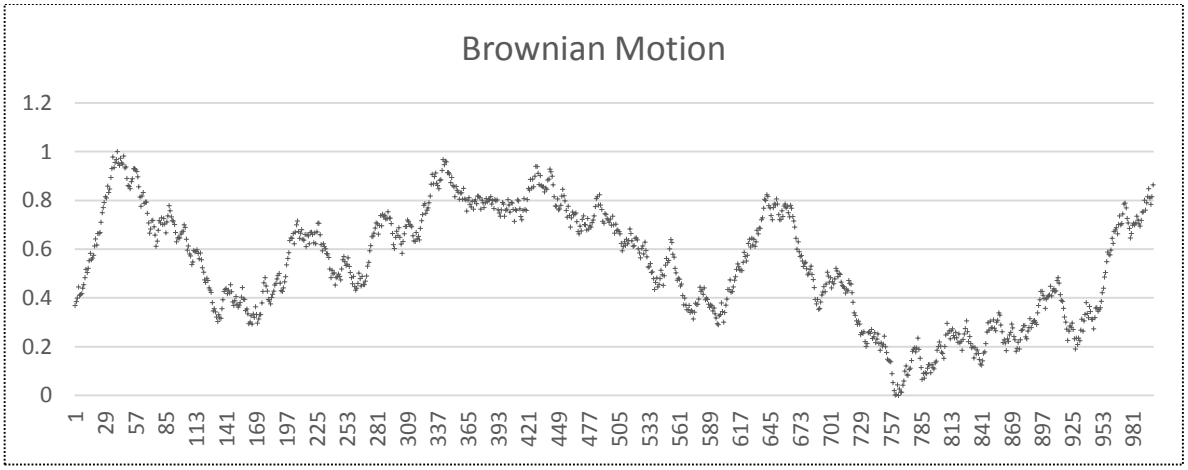
Powell, M. J. (1987). Radial basis functions for multivariable interpolation: a review. En *Algorithms for approximation* (págs. 143-167). New York: Clarendon Press.

Ramírez, J. (2014). *Estudio experimental de la aproximación de funciones con redes neuronales wavenets.* México.

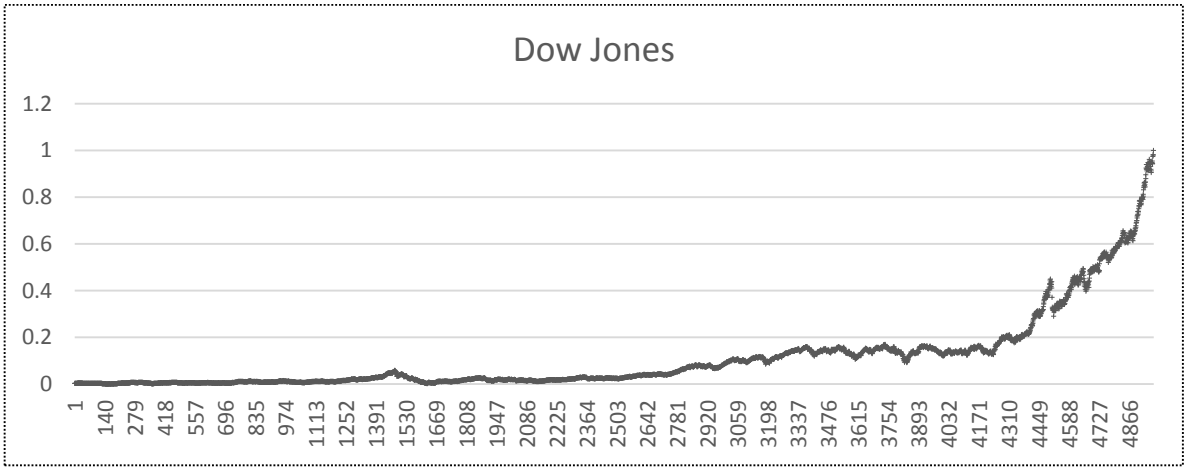
Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien.* Stuttgart.
Roach, D. L. (2002). Parameterizations of univariate orthogonal wavelets with short support. En *Approximation Theory XIII: San Antonio 2010* (págs. 369–384). Vanderbilt Univ. Press.
Romero, A. H. (2015). Aproximación con EPWavenets. *Research in Computing Science*, 95-109.
Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models.* New York: John Wiley and Sons.
Sitharama, L. S. (2002). *Foundations of Wavelet networks and application.* USA: Chapman and Hall.
Soman, K. R. (2010). *Insight Into Wavelets: From Theory to Practice.* Delhi: PHI Learning Private Limited.
Sotelo, C. (2002). The chemotactic hypothesis of Cajal: a century behind. . En *Progress in brain research 136* (págs. 11–20).
Storn, R. a. (1997). Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, 341-359.
Tewfik, A. a. (1992). On the Optimal Choice of a Wavelet for Signal Representation. *IEEE Transactions on information theory*, 747-765.
The MathWorks, I. (1994). *MathWorks*. Recuperado el 15 de Octubre de 2015, de http://www.mathworks.com/help/wavelet/index.html
Unidad Editorial Información Económica. (2016). *expansion.com.* Obtenido de http://www.expansion.com/diccionario-economico/dow-jones-industrial-average.html/
Wasilewski., F. (31 de Agosto de 2012). *Wavelet Browser*. Obtenido de http://wavelets.pybytes.com/wavelet/db1/
Weigend, A. G. (1994). *Time Series Prediction: Forecasting the Future and Understanding the Past.* Santa Fe: Reading, MA.
Weisstein, W. E. (20 de marzo de 2016). *"Logistic Equation." From MathWorld* . Obtenido de http://mathworld.wolfram.com/LogisticEquation.html
Widrow, B. a. (1990). 30 years of adaptive neural networks: perceptron, Madaline, and backpropagation. *Proceedings of the IEEE*, 1415-1442.
Wolpert, D. H. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 8*, 67-82.
Xudong, T. X. (2009). The Generalized Wavelets Based on Meyer Wavelet. En *Computational Science and Its Applications* (págs. 708-716). Seul: ICCSA.
Yates, R. (15 de Agosto de 2009). *Digital Signals Labs.* Obtenido de http://www.digitalsignallabs.com/white.pdf
Ypma, A. D. (1997). Using the Wavenet for function approximation. En *Proc. of ASCI'97* (págs. 236-240). Delft: The Netherlands.
Zhang, Q. B. (1992). Wavelet networks. En *IEEE Transactions on Neural Networks 3(6)* (págs. 889–898).

# Anexos

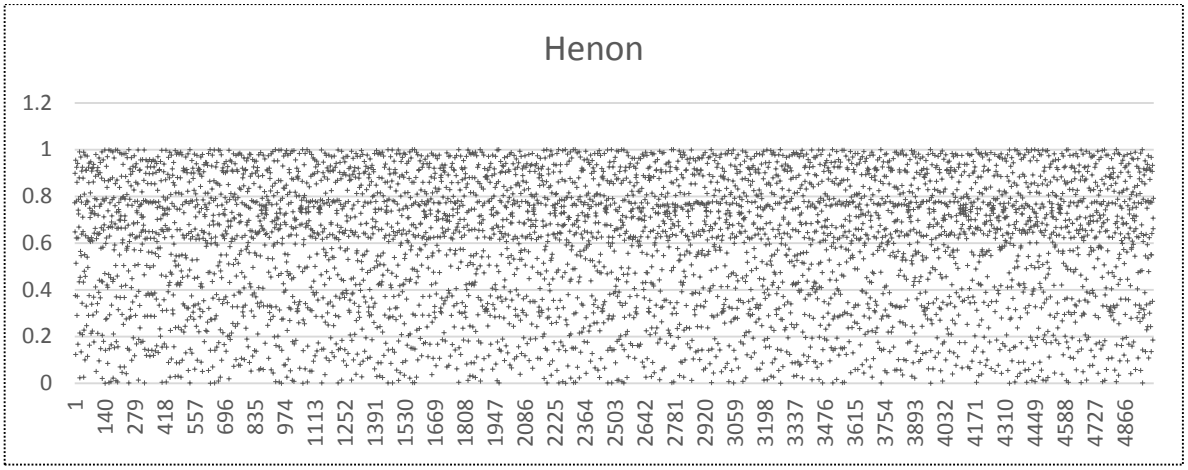
A. Funciones Objetivo



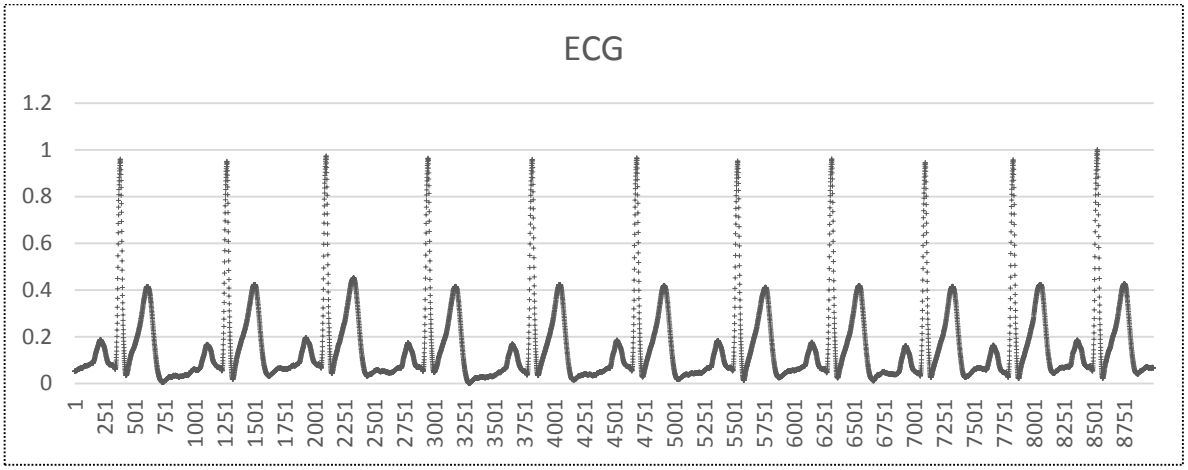

El movimiento browniano es la imagen macroscópica que emerge de una partícula que se mueve aleatoriamente en un espacio d-dimensional sin hacer muy grandes saltos, en este caso se trata el movimiento unidimensional en una versión discretizada (Deparment of statistics, 2008).



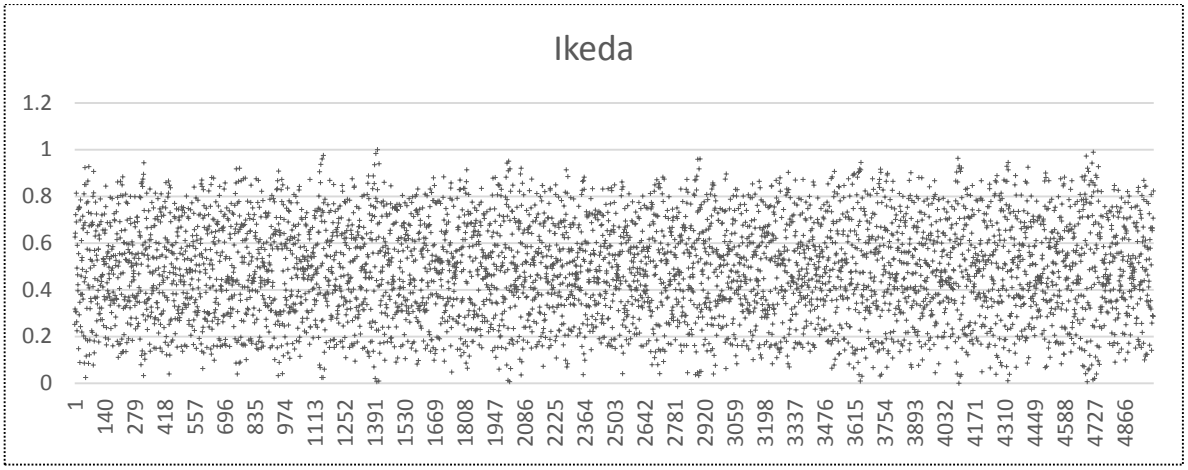

La función Down Jones fue creada en los años 90 por Charles Dow. El objetivo de esta función es dar una representación exacta del valor del mercado de Estados Unidos, con índice que es compuesto por 30 empresas industriales con mayor capitalización bursátil que cotizan en este mercado (Unidad Editorial Información Económica, 2016) .



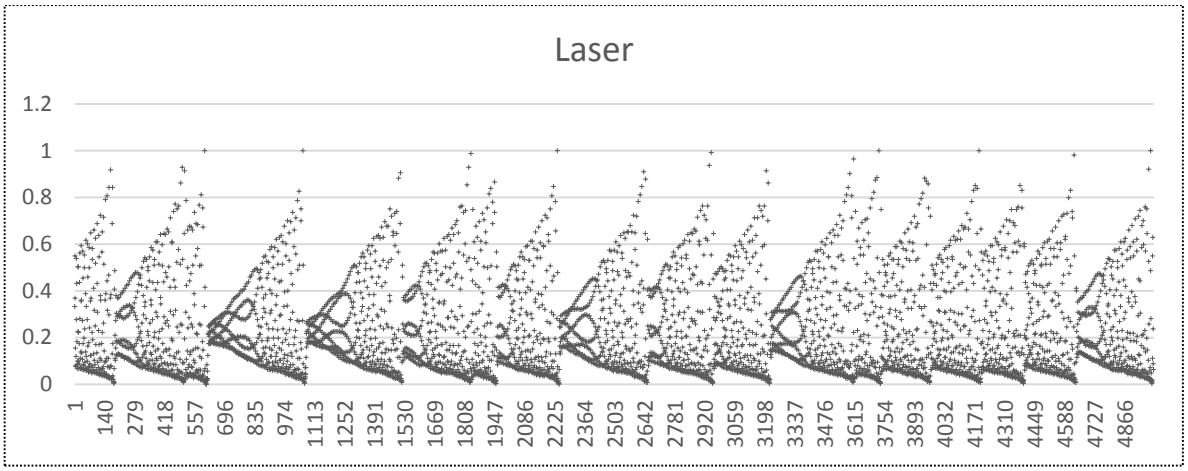

Esta función hace referencia al Mapa de Hénon que fue presentado en el artículo (Hénon, 1976).



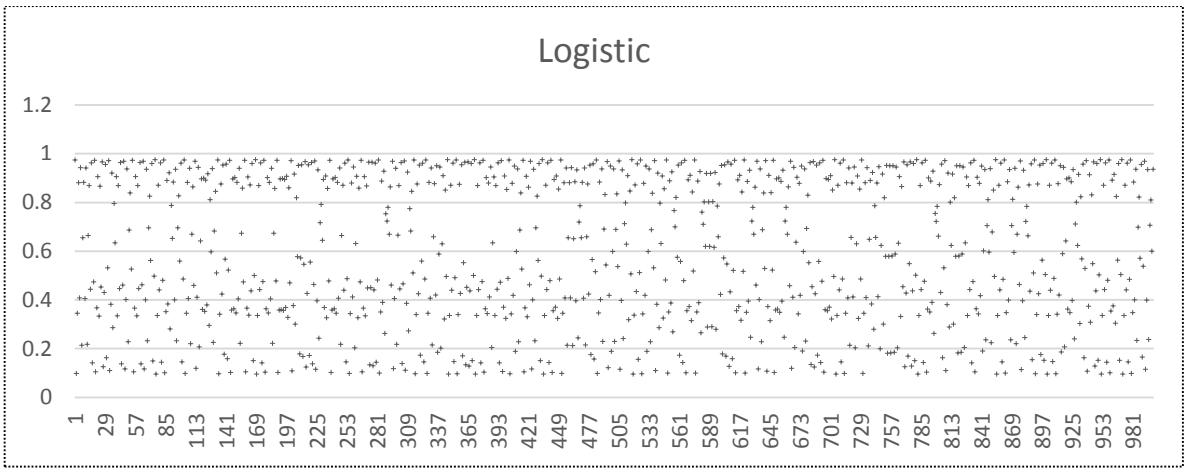

Esta función representa la actividad eléctrica del corazón. El pico principal es llamado onda R y corresponde a la contracción de los ventrículos. Un intervalo RR es el tiempo que existe entre dos ondas R consecutivas, y es inversamente proporcional al ritmo cardiaco (Cammarota, 2009).



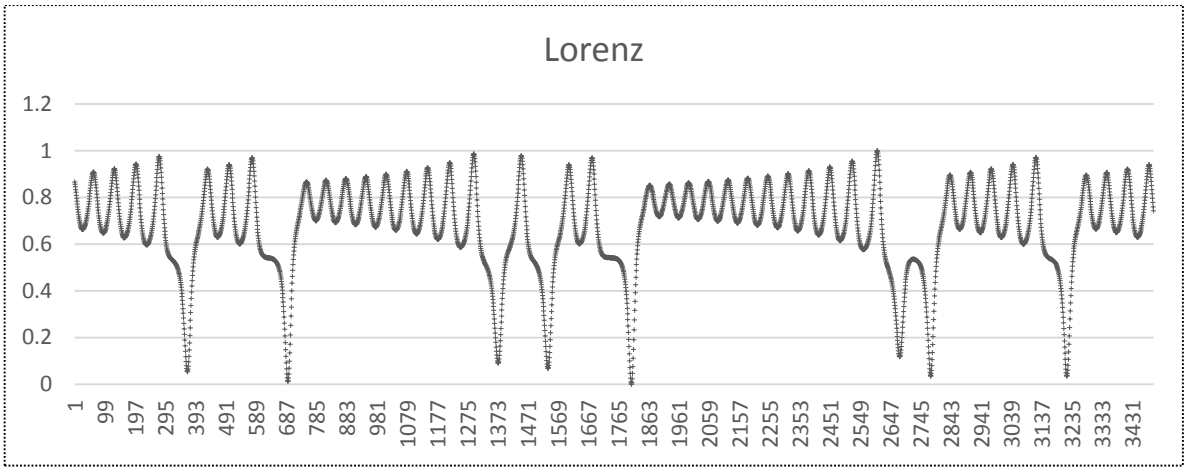

Esta función representa el mapa de Ikeda (Ikeda, 1979) el cual es un sistema dinámico de tiempo discreto. Es un modelo de luz que viaja a través de un resonador óptico no lineal.



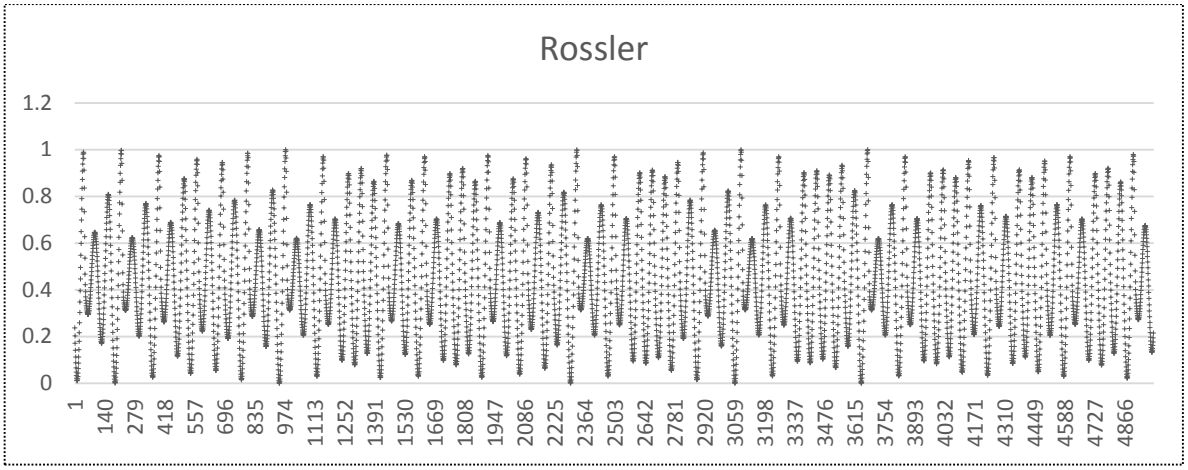

Esta función representa un registro de tiempo univariante de una simple cantidad observada. Los datos fueron aportados por Udo Huebner, Phys.-Techn. Bundesanstalt, Braunschweig, Alemania, y se recogieron principalmente por N. B. y C. Abraham O. Weiss. Estos datos se registraron a partir de una de infrarrojo lejano-láser en un estado caótico (Weigend, 1994).



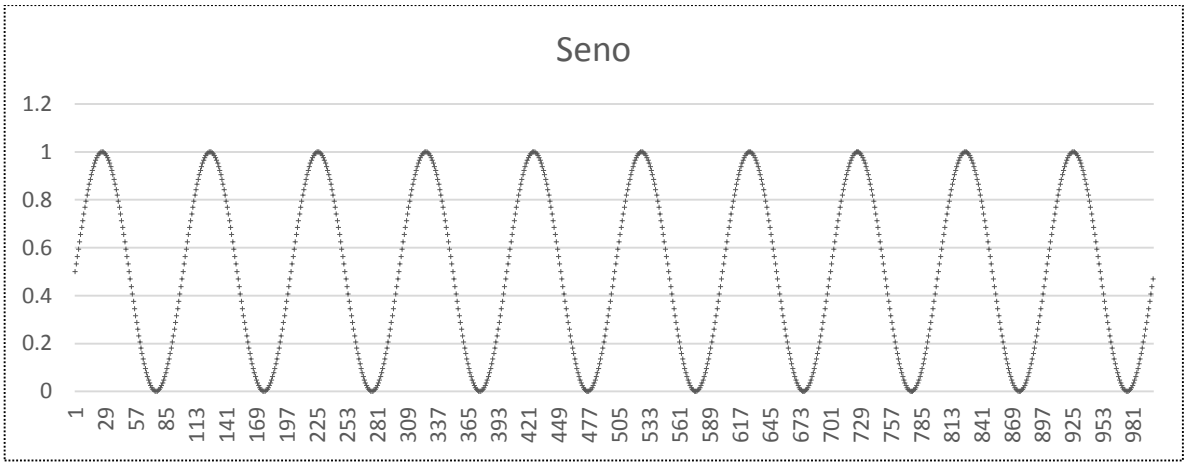

Esta función (también llamada curva de crecimiento logístico) representa un sistema caótico que es continuo en el tiempo. Esta función es generada por 3 simples ecuaciones dinámicas no lineales (Weisstein, 2016).



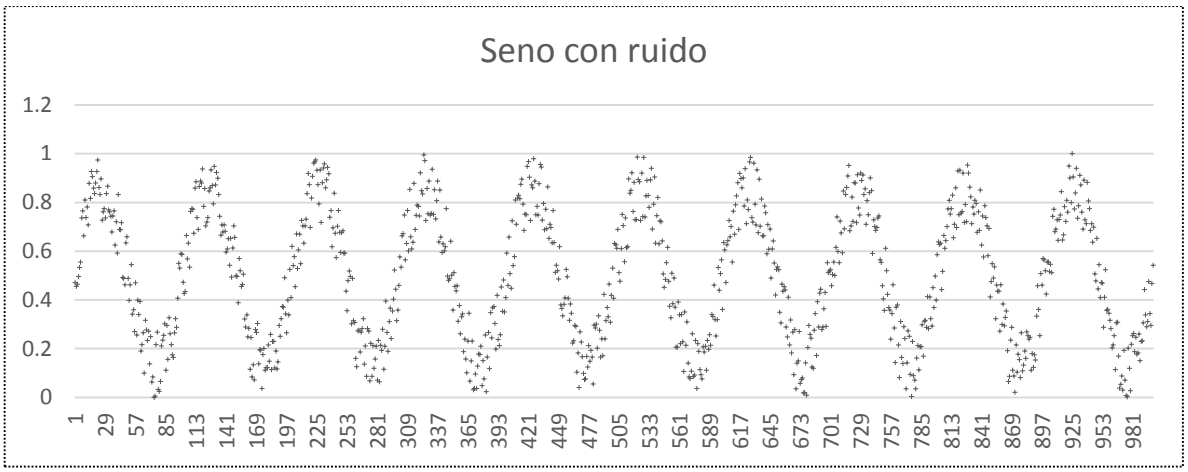

Esta función representa un sistema dinámico de tres variables, este sistema evoluciona con el tiempo y tiene un patrón que nunca se repite (Mancha, 2016).



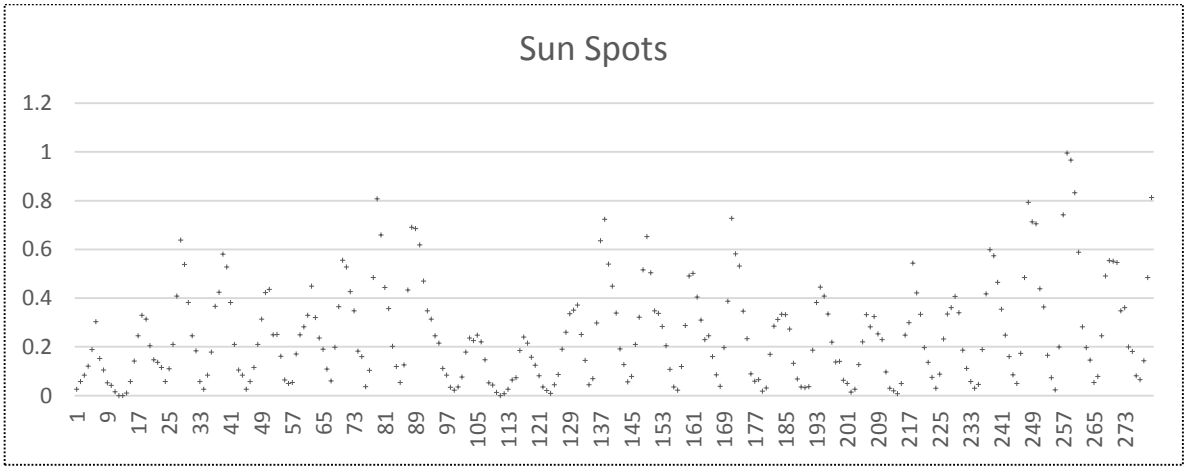

Esta función son tres ecuaciones ordinarias no lineales, las cuales definen un sistema dinámico continuo, además muestra las propiedades caóticas de los fractales (Mancha, 2016).



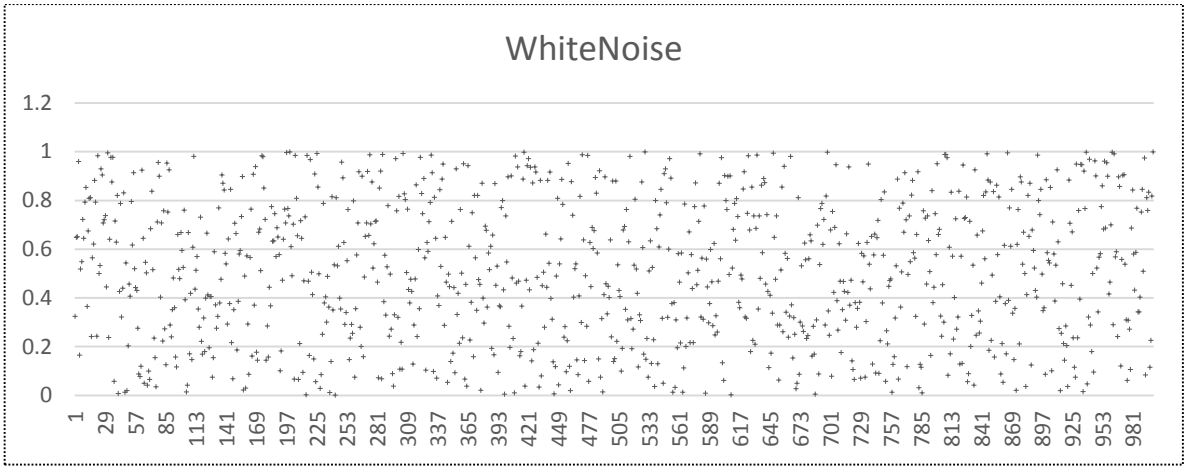

Es una función básica de la trigonometría que es continua y periódica en $2\pi$, donde $\theta$ es una ángulo que se mide en sentido contrario a las manecillas de reloj desde el eje $x$ a lo largo de un arco en un círculo.




Esta función es una variante de la función seno, con la diferencia que contiene ruido, es decir, pequeñas variaciones que distorsionan la señal.




Esta función es una serie natural que representa el registro anual de conteo de las manchas solares del año 1700 al año 1979.




Esta función representa una señal discreta donde los valores son una secuencia de variables no correlacionadas, con una varianza constante y una media constante (Yates, 2009).